# Tutorial - 05

Ques.1. what is difference between DFS and BFS. Please write the application of both algos.

→ using BFS, we can find the minimum no. of nodes between source node and destination node, while using DFS, we can find if a path exists between two nodes.

Applications of DFS -

i) Detecting cycles in a graph.
ii) Finding path between two given vertices u and v.
iii) If we perform, DFS on unweighted graph, then it will create minimum spanning tree for all pair shortest path tree.
iv) Topological sorting can be done using DFS.

Applications of BFS -

i) Like DFS, BFS may also be used for detecting cycles in graph.
ii) Finding shortest path and minimum spanning tree in unweighted graph.
~~Finding shortest path & minimum spanning~~
iii) Finding route through GPS navigator system with minimum no. of crossing.
iv) In networking, finding a route for packet transmission.

Ques.2. which Data structures are used to implement BFS and DFS ~~&~~ why?

DFS uses stack data structures as order doesn't has much importance.
BFS uses queue data structure as order matter's in this case.

Ques.3. what do you mean by sparse & dense graphs ? which representation of graph is better for sparse & dense graphs ?

→ sparse graph - Graph in which no. of edges is much less than the possible no. of edges.

Dense graph - Graph in which no. of edges is close to the maximal no. of edges.

If the the graph is sparse, we should store it as a list of edges. Alternatively, if it is ~~done~~ dense, we should store it as adjacency matrix.

Ques.4. How can you detect a cycle in graph using BFS and DFS ?

→ Using BFS:

1) compute in degree(no. of incoming edges) for each of the vertex present in graph and count no. of nodes = 0

11) pick all vertices with indegree as 0 and add them to queue.

111) Remove a vertex from the queue, then increment count by 1 and decreases in degree by 1 for all neighbours. If in-degree of a neighbouring node is 0, add to queue.

IV) Repeat step 3 until queue is empty.

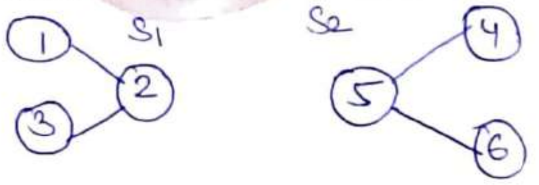
V) If no. of visited nodes is not equal to no. of nodes, then graph has a cycle.

→ Using DFS:
similar process is done is DFS as well, but in DFS, we ha the option of doing recursive called for vertices which are adjacent to the current node & are not get visited. If recursive func. returns false, then graph does not have a cycle.

Ques.5. what do you mean by disjoint set data structure? explain 3 operations along with examples, which can be performed on disjoined sets.

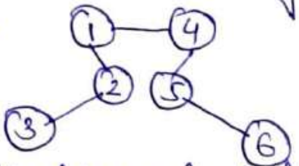
→ It allows to find out whether the 2 elements are in the same set or not efficiently.
The disjoint set can be divided as the subsets where there is no common element between 2 sets.

eg. - $S_1 = \{1, 2, 3\}$
$S_2 = \{4, 5, 6\}$



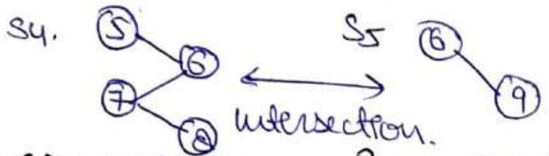
operations -

i) union - Merge 2 sets when edge is added.
$S_1 \cup S_2 = S_3 \rightarrow$

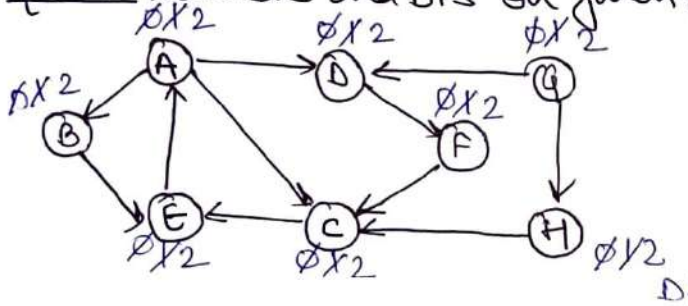


ii) find() - tells which element belongs to which set
find(1) = S1 , find(5) = S2

iii) intersection - outputs another set as common elements
$S_1 \cap S_2 = \{\Phi\}$ , $S_4 \cap S_5 = \{6\}$



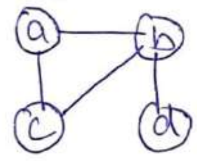
Ques.6. Run BFS and DFS on given graph.

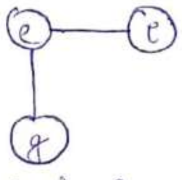


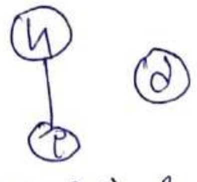
BFS:- nodes G H F D C E A B
parent × G G G H C E A
visited nodes → G H F D C E A B
path → G→H→C→E→A→B

DFS: nodes processed G G D C E A B
stack G DFH CFH EFH AFH BFH FH
path → G→B→C→E→A→B

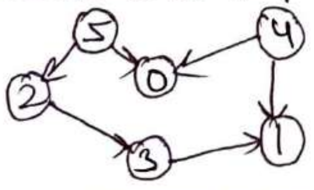
Ques.7 Find out no. of connected components & vertices in each comp. using disjoint set data structure.

(I)

no.(V) = 4
no.(CC) = 1

II)

no.(V) = 3
no.(CC) = 1

III)

no.(V) = 3
no.(CC) = 2

Ques.8. Apply Topological sorting & DFS on graph starting char(p) vertices 0 to 5.



Adjacency list
0 →
1 →
2 → 3
3 → 1
4 → 0, 1
5 → 2, 0

stack | 0 | 1 | 3 | 2 | 4 | 5 |
Topological: 5, 4, 2, 3, 1, 0
DFS stack → | 4 | 0 | 1 | 3 | 2 | 5 | Head →
DFS → 5 → 2 → 3 → 1 → 0 → 4

Ques. 9. Heap data structure can be used to implement priority queue? Name few graph algos where you need to use priority queue & why?

We can use heaps to implement priority queue. It will take O(logn) time to insert & delete each element in the priority queue. Based on heap structure, priority queue has also 2 types → max & min. priority queue. some algos. where we need to use priority queue:

i) Dijkstra's shortest path algo using priority queue - when graph is sorted in the form of adjacency list or matrix, priority queue can be used extract minimum efficiently when implementing Dijkstra's algorithm.

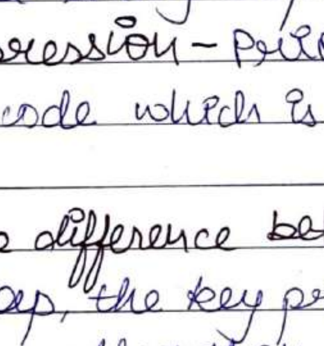
ii) Prim's algorithm - Priority queue is used to implement prim's to store key's of nodes & extract minimum key node at every step.

iii) Data compression - Priority queue is used in Huffman's code which is used to compress data.

Ques. 10. What is the difference between max & min heap?

In min heap, the key present at the root node must be smaller than among the keys present at all of its children.

eg.-



In max keep, the key present at the root note must be greater than among the key present at all of its children.