# Breadth-First Search (BFS)



In this strategy we explore the neighbors of a node before going deeper. We use the same adjacency structure as in DFS.

The time stamping is a little different. A vertex's neighbors get one more in timestamp than the vertex itself.

for $i := 1$ to $n$ do $V[i].d := -1$ endfor;
$V[s].d := 0$; $V[s].\pi :=$ nil; $Q := \{s\}$;
BFS(s);

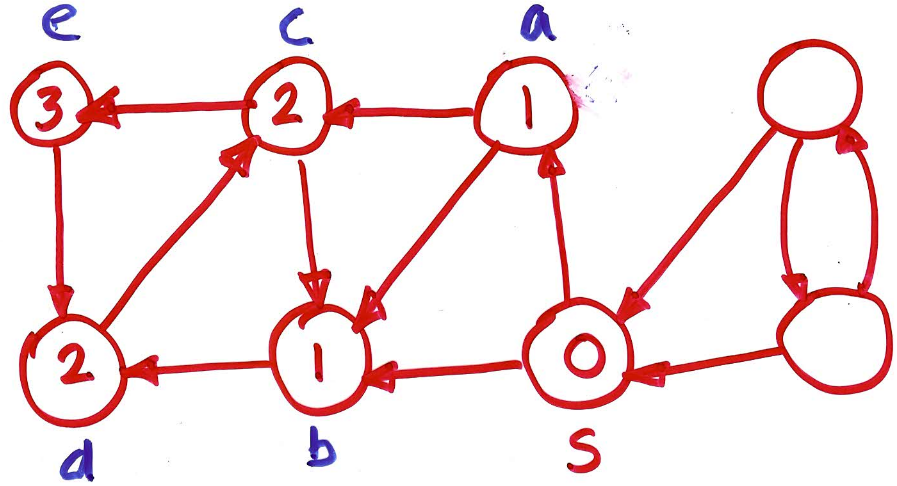
BFS uses a queue $Q$ which is initialized to a vertex $s$ from which search begins.

```
Procedure BFS(s);
 While Q ≠ ∅ do  j := DEQUE;  t := V[j].adj;
       While t ≠ nil  do
          if V[t.v].d = -1  then
                 V[t.v].d := V[j].d + 1
                 V[t.v].π := j;
                 ENQUEUE (t.v)
          endif
          t := t.next;
       endwhile
endwhile
```

$$\Uparrow$$

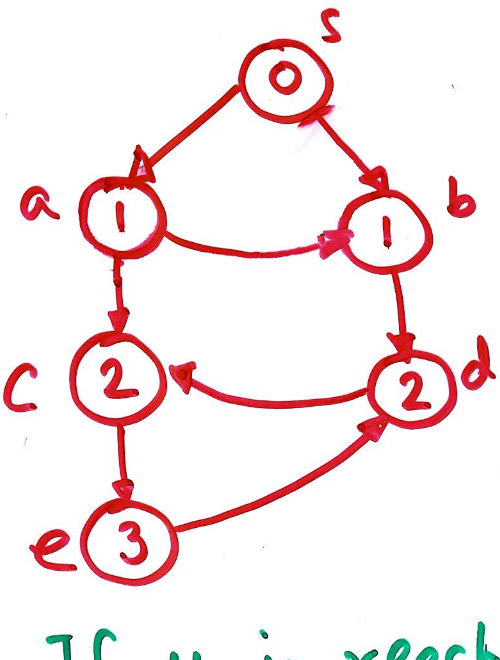
$$T(n, m) = O(n+m)$$

Ex.




~~s~~ ; ~~a~~, b, ~~c~~, ~~d~~, ~~e~~

The BFS tree can be redrawn which represents the part of the graph reachable from $s$.




For $u \in V$ define $\delta(s,u)$ as the number of edges of a shortest path from $s$ to $u$.

Claim. If $u$ is reachable from $s$ then $V[u].d = \delta(s,u)$ after completion of BFS.