Breadth first search

$G = (V, E)$

Start at a vertex s

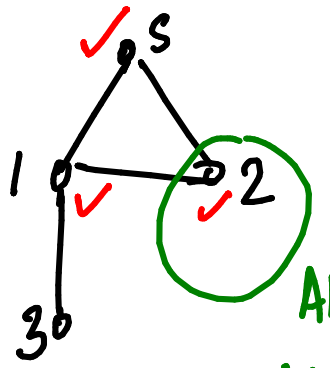
Mark s

Mark neighbours of s

Mark nbrs of nbrs of s

⋮

Till nothing more can be marked



Already marked by s, don't mark again from 1

Keep track of whether a marked node's neighbours have been explored

Want to ensure that each marked node is "explored" only once — termination

Suggestion: Keep a queue of marked but unexplored nodes
Each node enters queue when it is marked for the first time

Mark s, put it in queue
While queue is not empty
Extract head of queue as v
Explore nbrs of v, mark & add newly marked to queue

$s \in \{1,2..n\}$

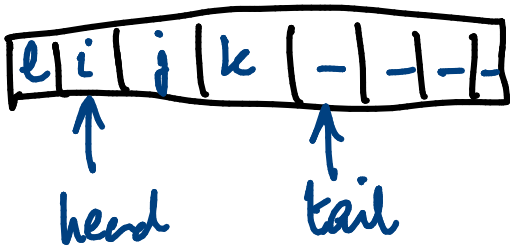
```
def bfs(A,s):
    for j in 1 to n:
        Mark[j] = 0
    Q = []
    Mark[s] = 1
    Q.append(s)
    while Q ≠ []:
        v = Q.extract_head()
        for j in 1 to n:
            if A[v][j] == 1 and Mark[j] == 0:
                Mark[j] = 1; Q.append(j)
```

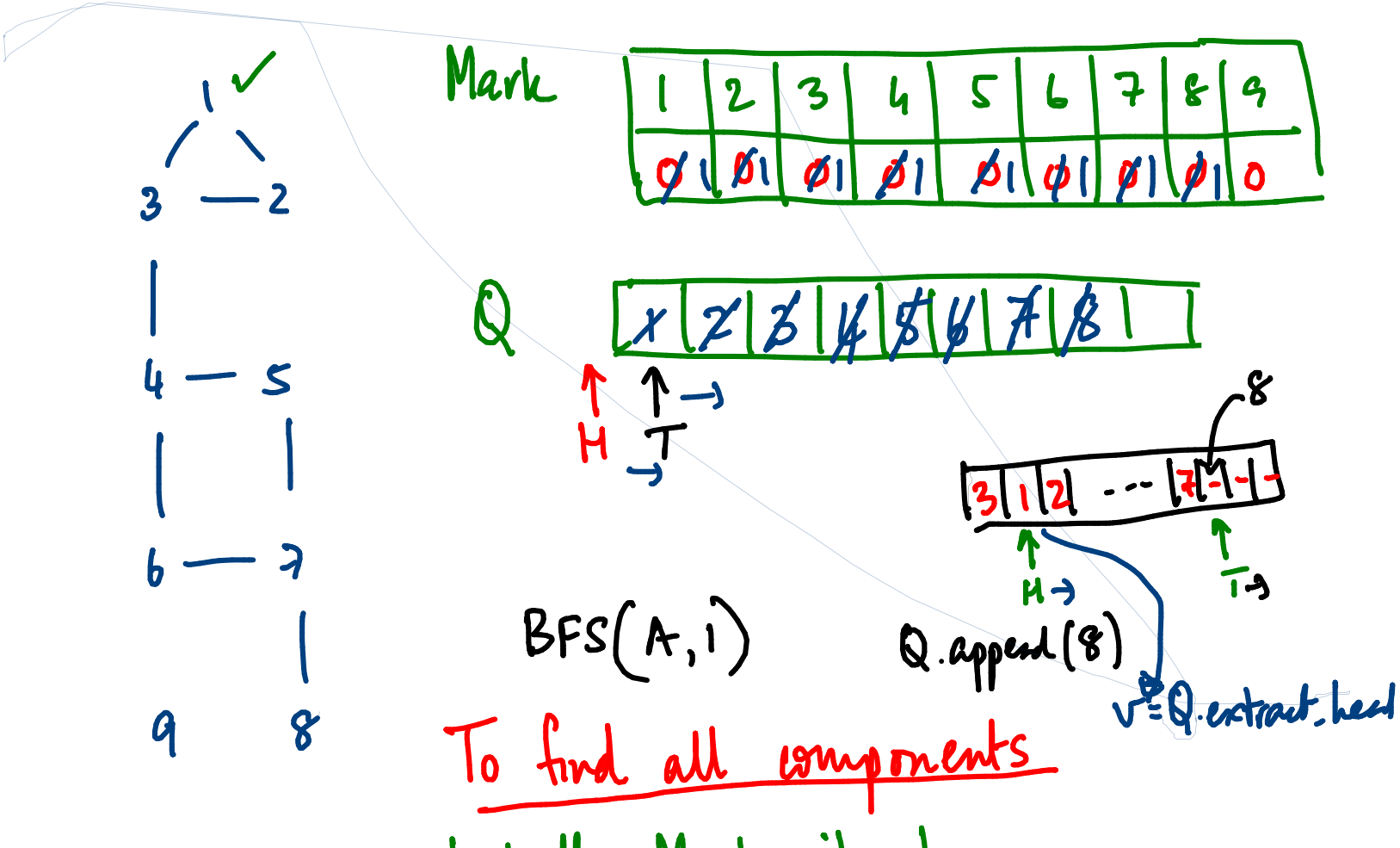
Assume $V = \{1,2,..,n\}$

Edges represented as adjacency matrix $A : n \times n$

$A[i][j] = 1$ iff $(i,j) \in E$

(symmetric if undirected)




1
3
2
4
5
6
7
9
8
Mark
1 2 3 4 5 6 7 8 9
0 0 0 0 0 0 0 0 0
Q
H
T
8
3 1 2 ... 7
H
T
BFS(A,1)
Q.append(8)
v=Q.extract_head
To find all components
Initially Mark with 1
Find first unmarked node, restart BFS
with mark 2, ...

Complexity?

$|V| = n$ $|E| = m$

Und: $m \leq \binom{n}{2}$

Dir: $m \leq n(n-1)$

Outer loop (for v in V)

for each neighbour of v

$O(n^2)$ Why is this n^2, even if $|E| = O(n)$?

No way to find neighbours except to scan a row of A

A is very sparse - most entries are 0

Instead, keep only useful information

Alternative representation of G

i → [neighbours of i]

1 → [2,3]

2 → [1,3]

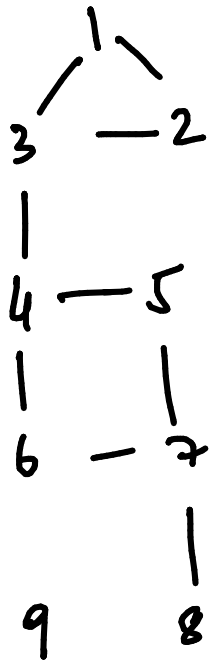
3 → [1,2,4]

⋮

8 → [7]

9 → []

ADJACENCY LIST



O(m) representation

Actually O(m+n)

to take into account disconnected portions

If we use adjacency list [List of lists]

Now, analyze BFS

while Q ≠ []

:

for each neighbour of v ← deg(v)

...

Total work is

$2m = deg(1) + deg(2) + \ldots deg(n)$

Overall : $O(m+n)$ LINEAR

	Process all neighbours	Is $(i,j) \in E$?
Adj Matrix	$O(n)$	$O(1)$
Adj list	$O(\deg(v))$	$O(\deg(v))$

BFS explores a graph "level by level"

Each node has a level - distance from start vertex

Shortest paths to vertices, in terms of no. of edges

Include this calculation within BFS

distance(v) for each v = min. no of edges from s to v

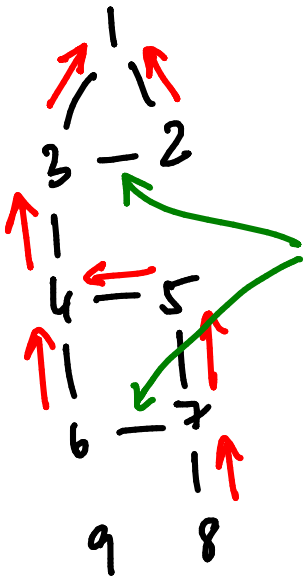
```
distance[s] = 0

while Q ≠ []
    :
    for each neighbour j of v
        if Mark[j] == 0
            Mark[j] = 1
            Q.append(j)
            distance[j] = distance[v] + 1
```

Mark[j] == 1 ⇒ j is reachable from s

But how?

Associate parent[j] for each j

If j is marked via v, set parent[j] = v



↑ = parent ⇒ ignore arrows : tree

edges in E, not in tree "Cross Edges" ⇒ G has a cycle

BFS Tree for G - not unique, depends on order of processing neighbours