Introduction to graph algorithms

Graph $G = (V, E)$

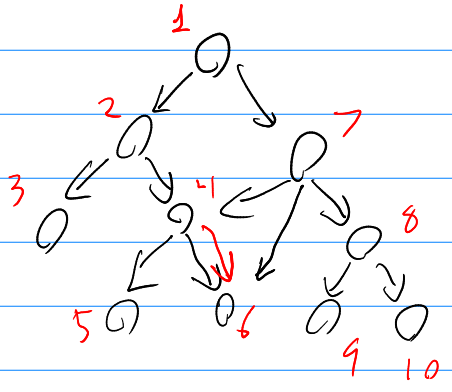
V = Set of vertices = # vertices
E = Set of edges = # edges

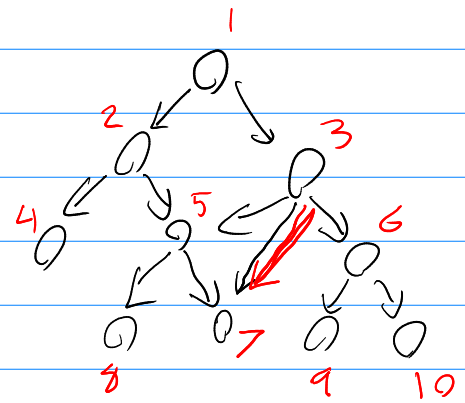
$E \leq \binom{V}{2}$ (undirected)
$\leq V(V-1)$ (directed)

Basic Question: Reachability

Given s, t find if $s \to t$ path exists
(maybe: & return path)

```
visited = {}
def DFS(v):
    if v in visited: return False
    visited.add(v)
    if v == t: return True
    for w in v.adj:
        if DFS(w): return True
```





```
def BFS(s, t):
    Q = queue([s])
    visited = {}
    while Q:
        v = Q.pop_front()
        if v in visited: continue
        visited.add(v)
        if v == t: return True
        for w in v.adj:
            Q.push_back(w)
    return False
```

DFS by Pop_back

(and loop in reverse order to match DFS() but both are depth first)

BFS: queue
DFS: stack

min. Spanning tree: heap / priority queue (Prim's Algorithm)
shortest paths: priority queue on different weights (Dijkstra's Alg)

Claim: whatever the pop() used, the whatever first search returns True $\Leftrightarrow \exists$ s→t path.

add Parent pointers to alg:

```
def BFS(s, t):
    Q = queue([(s, None)])
    Parent = {}
    while Q:
        v, P = Q.pop_front()
        if v in parent: continue
        Parent[v] = P
        if v == t: return True
        for w in v.adj:
            Q.push_back((w, v))
    return False
```

---

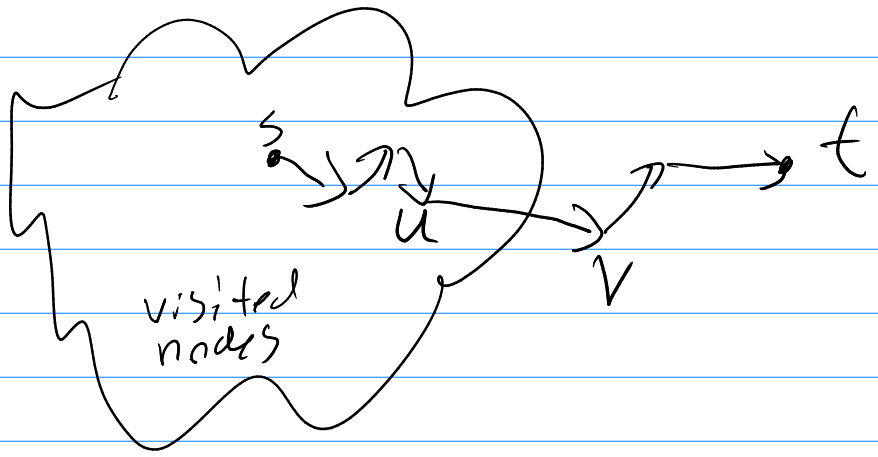
pf (⇒)

Consider the chain:

$t \to Parent[t] \to Parent[parent[t]] \to \cdots$

Except for (s, None), every (v, parent(v)) pair has a corresponding Parent[v] → v edge in E. Thus the chain either ends at s or in a loop. Every step of this chain moves to a vertex visited earlier in the execution, so it must terminate; hence it (backwards) would be an s → t path.

Pf ($\Leftarrow$) Suppose returns false but $\exists$ Path.

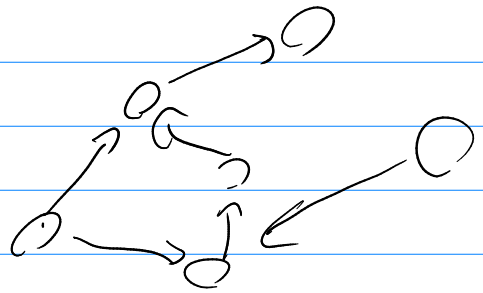


Then $\exists$ u visited, v not visited, u→v exists.
$u \neq t$.
When u visited, (v, u) added to Q
$\Rightarrow$ v would be visited before Q empty. $\Rightarrow \Leftarrow$.

BFS: Returns path w/ fewest steps

Exercises

1) Road network: weights = max height of trucks taking road (= min height of bridge)



Q1: tallest truck that can go $s \to t$?

Q2: can go between any pair of locations?

2) Floodfill: MS Paint