Introduction to graph algorithms

Graph $G = (V, E)$

$V =$ Set of vertices = # vertices

$E =$ Set of edges = # edges

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

$E \leq V(V-1)$ (directed)

Basic Question: Reachability

Given $s, t$ find all reachable vertices

(maybe: & return path)

$\text{visited} = \emptyset$

def DFS (v):

if $v$ in visited: return

visited.add(v)

for $w$ in $v$.adj:

DFS (w)
def BFS(s, E):
    Q = queue([s])
    visited = E
    while Q:
        v = Q.pop_front()
        if v in visited: continue
        visited.add(v)
        for w in v.adj:
            Q.push_back(w)

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 visits t = \text{last path.}
add parent pointers to algo:

```python
def BFS(s, E):
    Q = queue([s, None])
    Parent = dict()

    while Q:
        P = Q.pop_from()  # P
        if P in Parent:
            continue
        Parent[P] = None

        for w in P.adj():
            Q.push_back((w, P))

Claim: Parent[v] exists at end \( \iff \) all u.a. v visited

V reachable from \( S \), and if so, \( V \rightarrow \text{Parent}[v] \rightarrow \ldots \) ends at \( S \).

\( \text{Proof:} \ (V \text{ reachable } \iff \text{Parent}[v] \text{ exists}) \)

V reachable \( \iff \exists \) path \( S = u_1 \rightarrow u_2 \rightarrow u_3 \rightarrow \ldots \rightarrow u_n \equiv v \)
from \( S \) to \( V \).

we prove the claim \( V \) vertices at distance \( k \) from \( S \), by induction on \( k \).

Base case: \( k = 0 \).
Here, \( V = S \). The algorithm sets

\( \text{Parent}[S] = \text{None}, \) which exists.

Inductive step: Suppose true for \( k - 1 \).
Then \( \forall v \) of distance \( k \),

\( \exists \) path \( S = u_1 \rightarrow u_2 \rightarrow \ldots \rightarrow u_k \rightarrow u_{k+1} \equiv v \)
\( u_k \) has distance \( k - 1 \) \( \iff \) by inductive hypothesis it was visited.
When \( u_k \) was visited, since \( (u_k, v) \in E \).
(v, u_n) was placed in Q.

\( \Rightarrow (v, u_n) \) eventually removed from Q
\( \Rightarrow v \) visited at some point
\( \Rightarrow \text{parent}(v) \) set
\( \Rightarrow \text{inductive step true} \)
\( \Rightarrow \text{claim holds} \forall k \geq 0 \), as desired.

\( \text{Pr}(v \rightarrow \text{parent}(v) \rightarrow \cdots \text{ ends at } s) \)
\( \forall v \) "visited"

We prove this by induction on the # vertices visited.

Base case (1st vertex visited): \( v = s \), \( \text{parent}(s) = \text{None} \), \( \checkmark \).

Inductive step: If \( v \) is the \( k \)th vertex visited,
for \( k > 1 \), \( u = \text{parent}(v) \) was visited earlier,
so by the inductive hypothesis
\( u \rightarrow \text{parent}(u) \rightarrow \text{parent}(\text{parent}(u)) \rightarrow \cdots \)
\( \text{ends at } s \).

\( \Rightarrow v \rightarrow \text{parent}(v) \rightarrow \cdots \text{ ends at } s \).
\( \Rightarrow \text{inductive step} \Rightarrow \text{induction holds} \forall k \geq 0 \), claim.

\( \text{Pr}(v \text{ unreachable} \Rightarrow \text{parent}(v) \text{ does not exist or end}) \)

The previous proves the contrapositive:
If \( \text{parent}(v) \) exists \( \Rightarrow \text{end} \)
\( \Rightarrow v \rightarrow \text{parent}(v) \rightarrow \cdots \text{ ends at } s \)
each of them is a reverse edge in \( E \)
\( \Rightarrow \exists s \rightarrow v \text{ path} \)
\( \Rightarrow v \text{ reachable.} \)
Exercises

1) Road networks. Weights = max height of trucks taking road
of min height of bridge

Q1: tallest truck that can go S→E?
Q2: can go between any pair of locations?

2) Floodfill: MS Paint