CS124 Lecture 4

Feb 5, 2020

$G = (V, E)$ — $|V| = n$, $|E| = m$. Assume $m \geq n$, connected graph

$E \subseteq V \times V$

undirected $\{u, v\}$ or $(u,v) \in E \longleftrightarrow (v,u) \in E$

$f: V \to \mathbb{R}$, $E \to \mathbb{R}$

Model

- Transport Network
  - Quickest Route
  - Shortest Path
  - Cheapest Path
- Social Network
  - Spread of Disease/Information
- Scheduling
  - Determining conflict
  - SET

Representation

- Adjacency Matrix

$$\begin{bmatrix} & & \\ & \cdot & \\ & & \end{bmatrix} \qquad a_{ij} = \begin{cases} 1 & \text{if } i,j \text{ is edge} \\ 0 & \text{o/w} \end{cases}$$

can have functions

most useful for Directed, dense

Is $(i,j) \in E$? $O(1)$

What are the edges connected to i? $O(n)$

Space: $O(n^2)$

- Adjacency List

$1 \rightarrow \boxed{4} \rightarrow \boxed{3}$

$2 \rightarrow \ldots$

useful for sparse graphs

Is $(i,j) \in E$? $O(\deg(i)) = O(n)$

What are the edges connected to i? $O(\deg(i))$

Space? $O(m+n) = O(n^2)$

Graph Search Algorithms: Depth First Search

- adj list representation

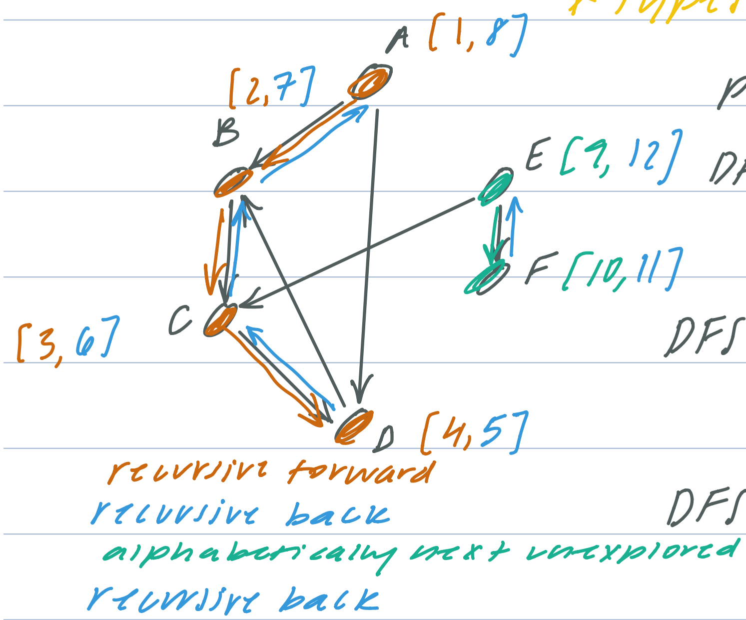
- Maze search w/ local view (adj list of edges)
- Stack-based approach

Algo:

```
Search(v):
    explored(v) := -1
    previsit(v)
    for (v,w) ∈ E:
        if explored(w) = 0
            search(w)
    postvisit(v)
```

```
DFS(V,E):
    for v ∈ V:
        explored(v) := 0
    for v ∈ V:
        if explored(v) = 0
            search(v)
```

* types of edges are order-dependent



previsit/postvisit: maintain counter

DFS: Tree Edge (edge followed during search)

AB, BC, CD

DFS: Back Edge (edge from node to a DFS ancestor

DB

DFS: Forward Edge (edge from node to descendant)

AD

DFS: Cross Edge (edge to a cousin, across branches)

EF

Claim: I_u, I_v are disjoint or one contains the other

$[pre(u), post(u)] = I_u$

$[pre(v), post(v)] = I_v$

Claim: If $(u,v) \in E$ then $post(u) < post(v) \longleftrightarrow (u,v)$ is a back edge.

Pf: By definition, v was on stack before u, v won't complete before u.

Case 1: I_v, I_u disjoint

u put on stack first, and finishes first.

Contradiction.

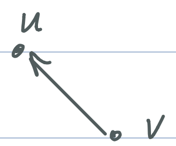
Case 2: $I_u \subseteq I_v$.

v put on stack first. $\exists$ path from v to u.

u gets put on stack. $\rightarrow (u,v)$ is a back edge.

Claim: $G(V,E)$ has a cycle $\longleftrightarrow$ DFS gives a back edge.

Pf.

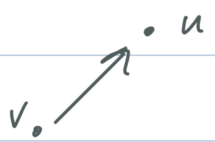


(v,u) gives back edge

by definition, path from u to v (tree edges)

and edge from v to u $\rightarrow$ cycle ■

Pf. Take the vertex w/ smallest postorder # in the cycle



by construction, postorder of u > postorder of v.

From previous claim, implies that (v,u) is a back edge.

Practical Question

- Processes that call each other, debug source before caller
- No cycles