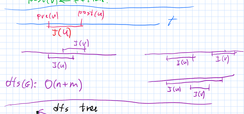
Lecture 19 10/25/22

DFS on directed graphs

time ← 0
dfs(u):
mark u as visited, pre(u) ← ++time
for all u → v do
if v is not visited then
dfs(v)
post(u) ← ++time



dfs(G): O(n+m)

Lemma

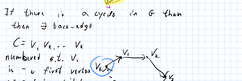
Let G be a directed graph
G has a cycle ⟺ there is a back-edge

Proof

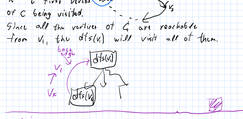
If there is a back edge

If there is a cycle in G then
then ∃ back-edge

C = $v_1, v_2 \dots v_k$
numbered s.t. v_1 is the first vertex visited or C being visited.



Since all the vertices of C are reachable from v_1, the dfs(v_1) will visit all of them.



Claim

Let G be a DAG.
If post(v) > post(u) then u → v is not an edge in G.

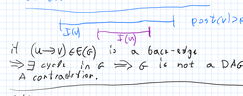
list vertices of the graph in decreasing post numbering:

$v_1 \; v_2 \; v_3 \; \dots \; v_k \; \dots \; v_n$

$post(v_1) > post(v_2) > \dots > post(v_k)$

Proof G is a DAG and post(v) > post(u)

a contradiction

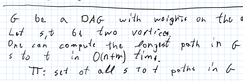


post(v) > post(u)

if (u → v) ∈ E(G) is a back-edge
⇒ ∃ cycle in G ⇒ G is not a DAG.
A contradiction.

G is a DAG:
Topological_sort(G):
DFS(G)
If there is a back-edge then throw a tantrum
list vertices in decreasing post numbering
"indexes" the desired numbering of the vertices
Output the numbering

O(n+m)



G be a DAG with weights on the edges
Let s, t be two vertices
Can compute the longest path in G from s to t in O(n+m) time.

Π: set of all s to t paths in G

$\arg\max_{\pi \in \Pi} w(\pi)$ $\quad w(\pi) = \sum_{e \in E(\pi)} w(e)$

Topological sort + DP

- Topological sort G $\quad$ O(n+m)

D[i]: The length of the longest path from v_i to $t = v_n$.

Init D[i] ← −∞ ∀i, D[n] ← 0.

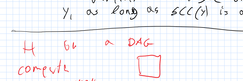
$D[i-1] = \max_{\substack{v_{i-1} \to v_k \\ k > i-1}} \left(w(v_{i-1} \to v_k) + D[k] \right)$

O(out_degree(v_{i-1}))

$\sum_{v \in V(G)} O(\deg(v)) = O(m)$

O(n+m)

SCC



scc(u) = reach(u, G) ∩ reach(u, G^{rev}) $\quad$ O(n+m)

O(n(n+m))

meta-graph G^{scc}

meta-graph is a DAG! $\quad$ O(n+m)

G^{scc}

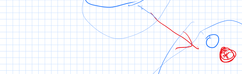
$k \in S_1$ $\quad$ dfs(k)

y in SCC of G that is a sink in G^{scc}.

dfs(y) = visit only the vertices in the SCC of y_1 as long as SCC(y) is a sink.

H be a DAG

compute a source of H



topological sorting (H^{rev})

Claim

list (G^{rev} numbering) in decreasing order.

For i = 1 to n do
if v_i is not taken,
SCC(v_i) ← dfs(v_i)

O(n+m)