Thursday April 16, 2015

Last class:

DAG = directed acyclic graph

Topological ordering = order vertices of a DAG so that all edges go left → right (low → high)

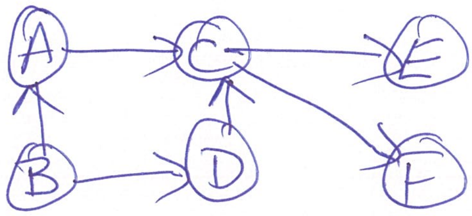
Key: for a DAG, its DFS tree has no back edges

Thus for all edges in a DAG, for $\vec{vw} \in E$, $post(v) > post(w)$

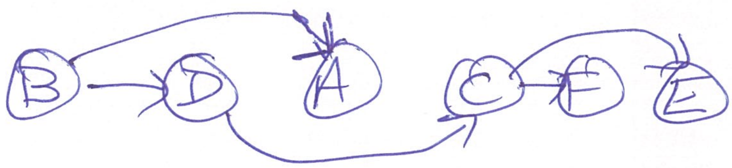
Topological sorting algorithm:

Run DFS & order by ↓ postorder #.

Example:



Topological ordering:



Source vertex = vertex with no incoming edges

Note 1^{st} vertex in topological ordering must be a source.

Sink vertex = vertex with no outgoing edges

Last vertex in top. ord. must be a sink.

So every DAG has ≥ 1 source & ≥ 1 sink.

Alternative topological sorting algorithm:

(1) Find a sink, output it & delete it

(2) Repeat (1) until the graph is empty.

Builds ordering right to left.

Can do with a source instead & build left → right.

Note: In a DAG,

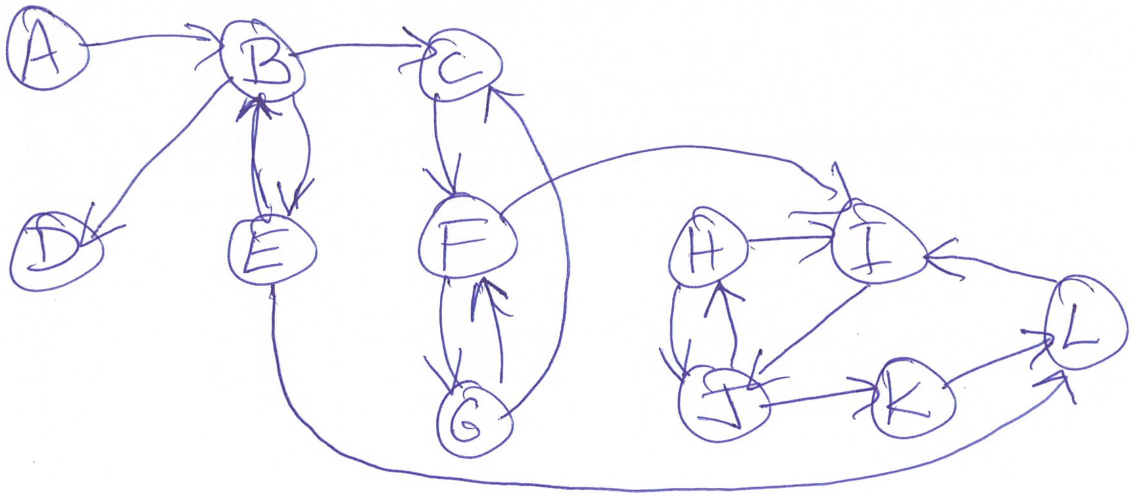
lowest post # is a sink

highest post # is a source.

Consider a general directed graph (may have cycles)

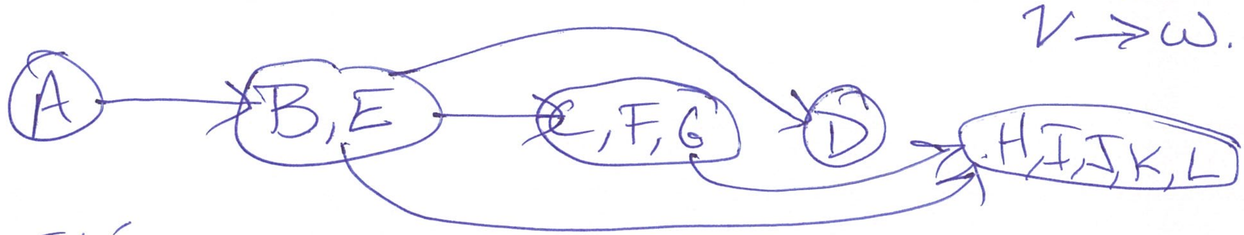
Vertices ~~v~~ v & w are strongly connected if: there is a path v to w & a path w to v

SCC = strongly connected component = maximal set of strongly connected vertices.



SCCs are {A}, {B, E}, {C, F, G}, {D}, {H, I, J, k, L}

Make a meta-vertex for each SCC
add edge from SCC S to S′
if some $v \in S$ & $w \in S'$ have edge
$v \to w$.



It's a DAG.

$\boxed{\text{Every directed graph is a DAG of its SCCs}}$

Why?
If there's a cycle including SCCs S & S′
then S & S′ are strongly connected
So $S \cup S'$ is a SCC (not S, S′)

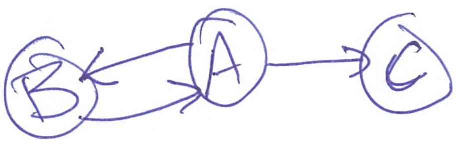
Today: Find SCCs & find topological ordering of SCCs.

Approach:
Find sink SCC, take it out & repeat

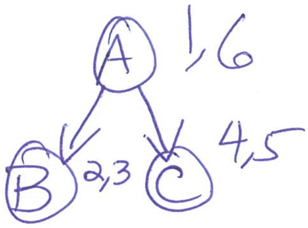
How to find a sink SCC?

If we have a vertex $v \in S$ where S is a sink SCC.
Run Explore(v) then we visit S and noone else.

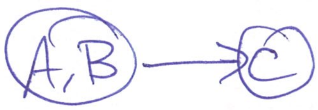
How do we find a vertex in a sink SCC?
Maybe vertex with lowest post #?
NO example:



DFS:



SCCs:



B has lowest post # but is not in a sink SCC.

But A has highest post # and is in a source SCC.
is it true in general? YES

<u>Lemma:</u> Vertex with highest Post # lies in a source SCC.

We can get a vertex in a <u>source</u> SCC but need a vertex in a <u>sink</u> SCC.

Look at G^R = reverse of G.

For $G=(V,E)$, let $G^R=(V,E^R)$

where $E^R = \{\overrightarrow{wv} : \overrightarrow{vw} \in E\}$

= reverse of every edge in G

Source SCC in G = Sink SCC in G^R

Sink SCC in G = Source SCC in G^R

Now we have our SCC algorithm.

SCC algorithm:

For input $G=(V,E)$,

1) Construct G^R.
2) Run DFS on G^R
3) Order V by decreasing post # from (2)
   (then 1st vertex is in a source SCC of G^R
   = sink SCC of G)
4) Run the (undirected) connected components algorithm on directed G.

DFS-cc(G)

For all $v \in V$, visited(v) = FALSE
cc = 0
for all $w \in V$ (where V is ordered by)
  if not visited(w) then:
    cc++
    Explore(w)

Explore(w):

visited(w) = TRUE
ccnum(w) = cc
for each $(w,z) \in E$:
  if not visited(z) then Explore(z).

Running time: $O(|V|+|E|) = O(n+m)$.

Proof of Key lemma:

(Vertex with highest post # lies in a source SCC)

Claim: if S & S' are SCCs
& if there is $v \in S, w \in S'$ where $\overrightarrow{vw} \in E$
then: max post # in S $>$ max post # in S'

Assuming the claim we can topologically sort the SCCs by the max post # in each SCC.
So SCC with max post # is 1st
hence it is a source SCC, and thus the vertex with max post # is in a source SCC. That proves the lemma.
We just need to prove the claim still.

For the claim:

Since there is a path $S \to S'$ (by $v \to w$;

there is no path $S' \to S$

otherwise $S \cup S'$ is a SCC.

Let z be the 1^{st} vertex in $S \cup S'$

that's visited by DFS.

If $z \in S'$ then we'll see all of S'

before seeing any of S, so

all post #'s in S' $<$ all post #'s in S

& the claim holds in this case.

If $z \in S$ then from Explore(z) we

see all of S & S' before

finishing z. So

Post # of z $>$ Post # of every $y \in (S \cup S') - z$

This proves the claim.

$\boxtimes$