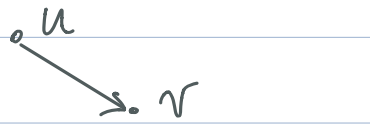
CS 124 Lecture 5

February 10, 2020

Topological Sort

Directed Acyclic Graph (DAG)



"u before v"

source: "INDEG" = 0

sink: "OUTDEG" = 0

Algorithm:

repeat
- Pick a source → Find degree 0 vertex
- Put it first
- Pull it and its edges out

Alternatively:

Do a DFS — O(m+n) time

Schedule by postorder # decreasing

Proof.

DAG → no backedges (3.2)

3.1
- $(u,v) \in E$
- $post(u) < post(v)$
- (u,v) is backedge

If $(u,v) \in E$, $post(u) > post(v)$

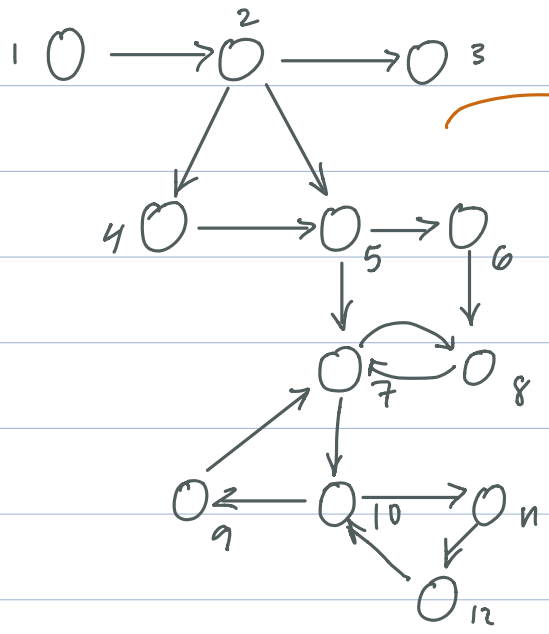
Otherwise, back edge by 3.1

Ordering w/ cycles

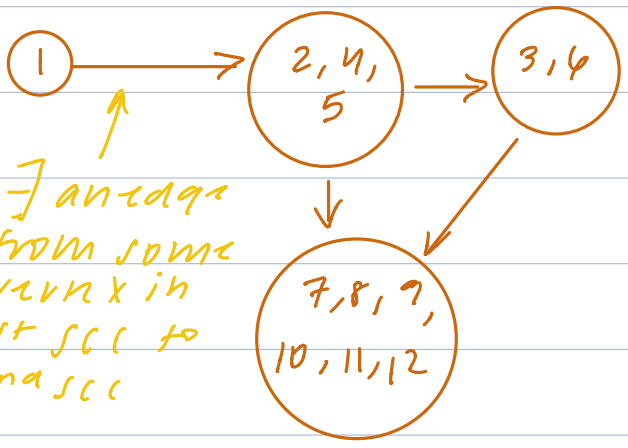
Strongly connected components

u, v in same strongly connected cycle if ∃ a path from v to v and v to u

SCC → DAG Example



collapse strongly connected components



∃ an edge from some vertex in 1st SCC to 2nd SCC

must be a DAG, otherwise would collapse

Computing SCCs

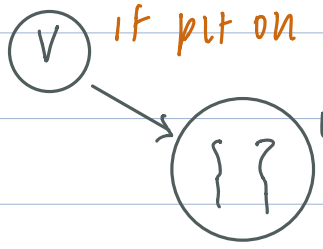
Property 1. If start DFS in a sink

Explore only the sink SCC, and then jump to a new component

reverse edges in graph

Property 2: Vertex with the highest postorder # is from a source SCC

Pf. By contradiction



if pt on before, explore and finish u before back to v

Algo.

DFS on G^R (reverse of the graph)

Get a vertex from sink SCC in G

Pull off sink SCC

Continue

By highest remaining postorder

Single DFS on original graph

Breadth First Search

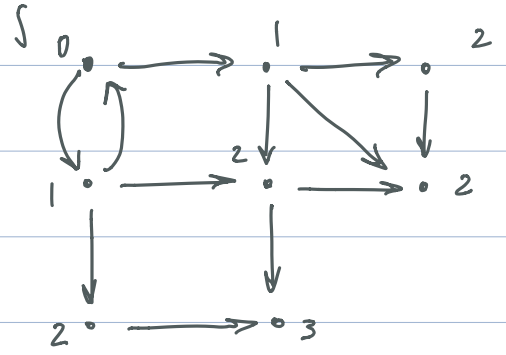
Queue, Base = 0

O(m+n) runtime

Done from startvertex(v).

```
dist[s] := 0
inject(q, s)
placed[s] := 0
```

```
while(size(q)) > 0:
    v := pop()
    for (v, w) ∈ E:
        if (placed(w) = 0):
            inject(q, w)
            placed(w) := 1
            dist[w] := dist[v] + 1
```

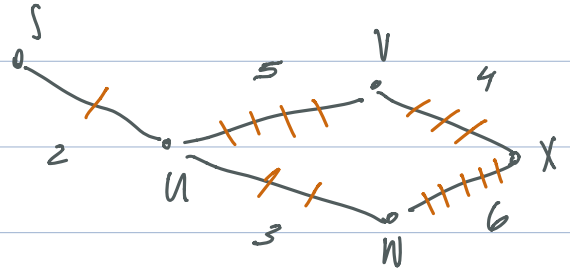


Claim: Shortest path lengths from s to other vertices.

- Invariant on distance d.
- All things of distance d get labeled d and all things of other distances do not

Single-source shortest path

- each edge has length
- reduction



add intermediate points spaced length 1 away

O(Σ length e)

Priority Queue

- Data structure: Heap

  deletemin(H) → give object of smallest value

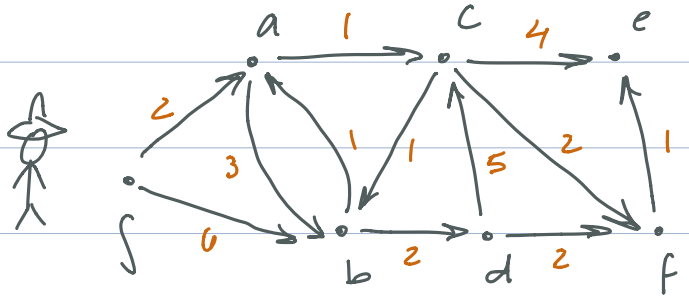
  insert(x, y, H) → insert x w/ value y

  change(x, y, H) → if x's value > y, replace x's value with y

Binary Heap

n objects in heap

all operations are O(log n)



Shortest path w/ + weights

Claims:

- Will get right shortest path