SSSP in DAGs (directed acyclic graphs)

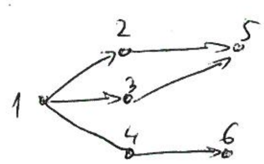
- DFS (depth first search)

```
dfs (vertex v) {
    v.visited = TRUE;
    for each w adjacent to v do
        if (! w.visited) then dfs(w);
}
```

- If $G = (V, E)$ not (strongly) connected ↦ may have DFS forest.

- Topological Sort: ordering of vertices in a DAG based on precedence: if path $v_i \overset{P_i}{\leadsto} v_j$ then v_i before v_j in ordering

  - not unique:

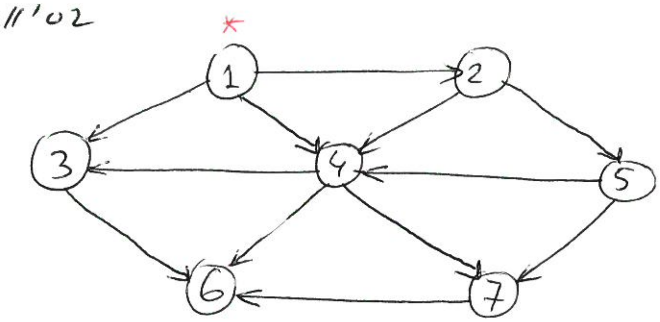


- $v \in V$: indegree (v) : # edges $(u, v) \in E$

- can use DFS to obtain TS order.

- Given directed graph G, is it a DAG?
  - can use TS to answer it (cannot find vertex with indegree 0)

E.g.:



TS algorithm:

- output (number) $v \in V$ with indegree 0
- (remove v and its outgoing edges)
- repeat until no vertex left

How to find $v \in V$ with indegree$(v) = 0$?

- use queue :
    - each time incomming edge at $u \in V$ is removed, decrease indegree(u)
    - if indegree(u) is 0, place u in queue
    - $O(|V| + |E|)$ time
- Init : • find $v \in V$ with indegree$(v) = 0$ in $O(|V|)$ time.
    • place in Queue.

vertex	Indegree before dequeue
1	0
2	1
3	2
4	3
5	1
6	3
7	2
Enqueue	
Dequeue	1

DAGs and TS

Lemma 1: A DAG has a sink and a source.

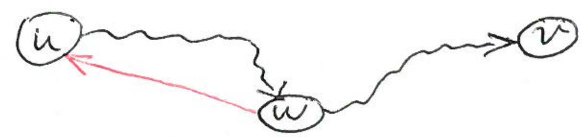
sink $\downarrow$ outdegree $= 0$ source $\downarrow$ indegree $= 0$

Proof:

Let P be the longest path in G, $u \overset{P}{\rightsquigarrow} v$.

Claim: u is a source.

If not, $\exists\ (w, u)$. If $w \notin P \Rightarrow P' = \{(w,u)\} \cup P$ is longer than P. If $w \in P \Rightarrow$ cycle:



A similar argument shows v is a sink.

Theorem 1: A directed G has a TS $\Leftrightarrow$ G is a DAG

Proof: Assume G has TS and a cycle C. Let $v_i \in$ be the vertex with smallest index assigned by T.
There is then an edge (v_j, v_i), $v_j \in C$, with $j > i \Rightarrow$ contradiction with TS assignmen

Now assume G acyclic $\mapsto$ use induction.

$\boxed{\text{Define } P(n): \text{a directed acyclic graph with } n \text{ vertices has a TS.}}$

$\Rightarrow P(1)$ true.

Assume $P(m)$ is true. Let G have $m+1$ vertices $\Rightarrow$ source v_0. $G \setminus \{v_0\}$ has TS $v_1, v_2, \dots, v_m \Rightarrow G$ has TS $v_0, v_1, \dots, v_n$

SSSP in DAGs (cont.)

Use vertex selection rule: select in TS order.

- SPs well defined, even if negative weight edges (negative weight cycles cannot exist)

```
DAG-SP(G, w, s)
    topologically sort vertices of G
    Init-Single-Source(G, s)
    for each u ∈ V, in TS order do
        for each v ∈ Adj[u] do
            Relax(u, v, w)
```

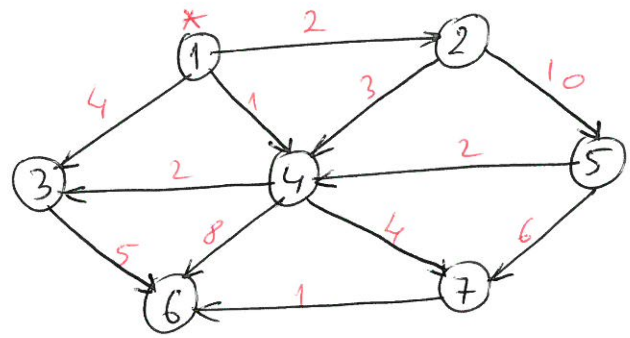
- $O(|V| + |E|)$ time.

- Critical path in a DAG: longest path through the DAG
  - Can find one by:
    (1) negating edge weights and running DAG-SP, or
    (2) run DAG-SP with modifications:
      - replace "∞" by "$-\infty$" in Init-Single-Source
      - replace "$>$" by "$<$" in RELAX

Note: Longest simple path in unweighted graph ↦ NP-complete

CS 6363 → Fall'02

Example DAG → SP:



Rule: $d_u = \min\{d_u, d_v + w(v,u)\}$

v	d_v	p_v
1	0	nil
2	~~∞~~ 2	~~nil~~ 1
3	~~∞~~ ~~4~~ 3	~~nil~~ ~~1~~ 4
4	~~∞~~ 1	~~nil~~ 1
5	~~∞~~ 12	~~nil~~ 2
6	~~∞~~ ~~9~~ ~~8~~ 6	~~nil~~ ~~4~~ ~~3~~ 7
7	~~∞~~ ~~18~~ 5	~~nil~~ ~~5~~ 4

Queue: ~~1~~, ~~2~~, ~~5~~, ~~4~~, ~~3~~, ~~7~~, ~~6~~