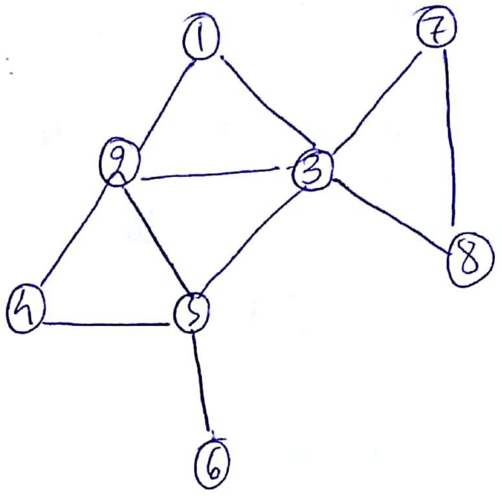
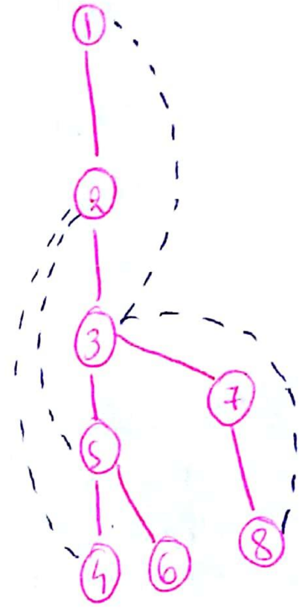
EE 360C - ALGORITHMS - FALL 2018 - Sep 27

G:



a DFS tree T:



n = # of nodes , m = # of edges

DFS using:

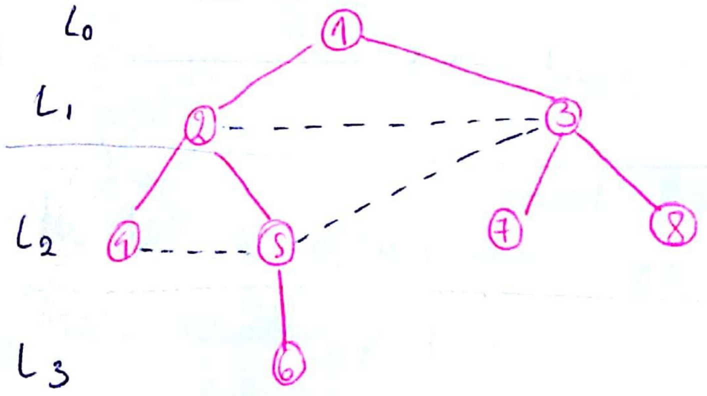
Adjacency list: $O(m \cdot n)$

Let T be DFS search tree.

Claim: Suppose edge (x, y) is in G but not in T.

Then one of x or y is an ancestor of the other in T.

BFS tree starting at 1:



L_i = all nodes that do not belong in a previous layer, and that have an edge from L_{i+1} in G.

Def: Shortest path distance $d(s,v)$ from node s to v is the minimum # of edges in any path from s to v. (if no path exists, $d(s,v) = \infty$)

Claim.

For BFS: $v \in L_i$ implies $d(s,v) = i$

(starting at s)

Claim: Adjacent nodes in G are either in the same layer or in consecutive layers in T (BFS tree)

Proof of shortest path claim:

Suppose $v \in L_i$. Then $d(s,v) \leq i$

Can't be < since that requires an edge that skips a layer. (according to the claim above). ☐

Topological Sort:

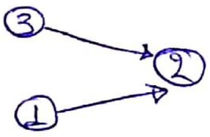
Set of tasks.
Precedence constraints: a set of pairs (v_i, v_j), meaning that v_i must occur before v_j

Can be represented by directed graph:

nodes: jobs

edges: precedence constraints

e.g.



3 before 2, 1 before 2.

<u>Def.</u> A topological order of a digraph is an ordering of its nodes as $v_1, v_2, \ldots, v_n$, s.t. for every edge (v_i, v_j), we have $i < j$.