DS 320 Algorithms for Data Science **Lecture #4 Worksheet**
Spring 2024 Prof. Kira Goldner

Depth-First Search

Graph search algorithms: good for exploring a graph, determining whether two nodes are connected, determining some properties regarding the ordered structure of a directed graph.

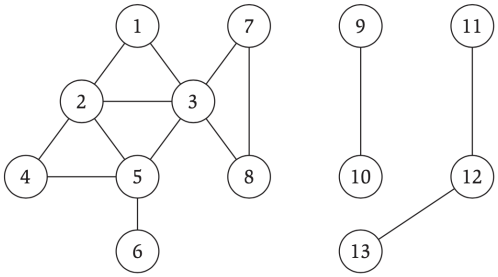
Depth-First Search: shoots as far away from a node possible to see if it results in a successful path, and only turns around if it dead ends.

Algorithm 1 DFS(G)

Input: Graph $G = (V, E)$ and vertex v.
for each $v \in V$ **do**
  v is unexplored
  set v's parent to Null
end for
for each $v \in V$ **do**
  search(v, G)
end for
return forest F formed by parents

Algorithm 2 search(v, G)

Input: Graph $G = (V, E)$ and vertex v.
mark v as explored
previsit(v)
for $(v, w) \in E$ **do**
  if w is unexplored **then**
    set w's parent to v
    search(w, G)
  end if
end for
postvisit(v)



Exercise.

1. Consider the pseudocode when called on the above example, that is, what happens when we run DFS(G) where G is the graph above? Draw the DFS forest as the graph is explored.

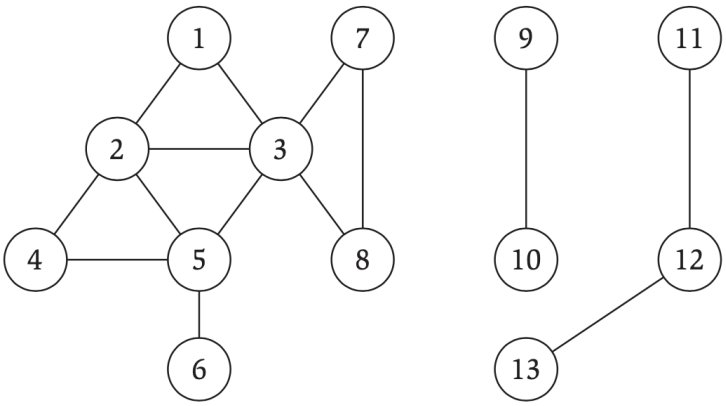
2. DFS implicitly uses a stack. Draw the stages of the stack as it runs on the example graph.

Preorder, Postorder, and Tree vs. Non-Tree Edges

Preorder and Postorder: Now, we'll define what happens in *previsit*() and *postvisit*(). Our idea is to track when nodes are pushed onto the stack and when they are popped off, as this order winds up giving us important properties. Formally, we have the following definitions.

Definition 1. The method previst(v) assigns a vertex v's *preorder* time, that is, i if it is pushed onto the stack at time i.

Definition 2. The method postvist(v) assigns a vertex v's *postorder* time, that is, i if it is popped off of the stack at time i.



We can partition the edges of G into four types:

- *Tree edges* are edges in the depth-first forest F. Edge (u, v) is a tree edge if was first discovered by exploring edge (u, v).
- *Back edges* are those edges (u, v) connecting a vertex u to an ancestor in a depth-first tree. We consider self-loops, which may occur in directed graphs, to be back edges.
- *Forward edges* are those nontree edges (u, v) connecting a vertex u to a descendant in a depth-first tree.
- *Cross edges* are all other edges. They can go between vertices in the same depth-first tree, as long as one vertex is not an ancestor of the other, or they can go between vertices in different depth-first trees.

Claim 1. If $(u, v) \in E$ then $postorder(u) < postorder(v) \iff (u, v)$ is a back edge.

Claim 2. $G = (V, E)$ has a cycle $\iff$ the DFS tree of G yields a back edge.

Application: Topological Sort

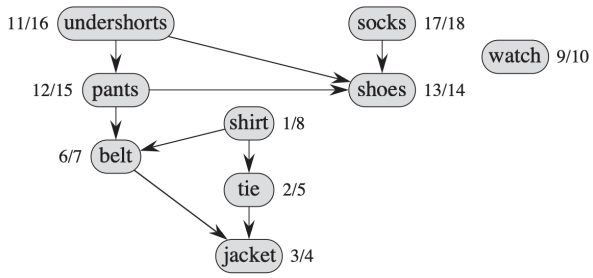


Figure 1: Top sort example graph from CLRS.

Theorem 1. *G has a topological order $\iff$ G is a DAG.*

Topological Sort Algorithm:

Theorem 2. *If the tasks are scheduled by decreasing postorder number, then all precedence constraints are satisfied.*