CSCE 411: Design and Analysis of Algorithms

Lecture 10: Graph Algorithms: BFS and DFS

Date: February 20 *Nate Veldt, updated by Samson Zhou*

Course Logistics

- Graph algorithms: Chapter 22
- Homework 4 out this weekend, due next Friday

1 Graph Representation

Consider a graph $G = (V, E)$ with a fixed node ordering $V = \{1, 2, \dots, n\}$.

Adjacency Matrix The **adjacency matrix $\mathbf{A}$** of G is defined so that

Adjacency List The **adjacency list** Adj of G is

Graph Activity

Consider the following graph:

- Write down the adjacency matrix
- Write down the adjacency list
- Write down the degree of each node
- Write down the neighborhood of node 3
- Find the number of connected components

Question 1. *Assume that $G = (V, E)$ is a graph in which each node has at least one edge touching it. Let $n = |V|$ and $m = |E|$. How much space is needed to store the graph?*

A $O(n)$

B $O(m)$

C $O(n^2)$

D $O(mn)$

2 Breadth First Search

Shortest Path Problem: Given a graph $G = (V, E)$ and source node $s \in V$, find the shortest path from s to every other $v \in V$.

We will do this using the *breadth first search* algorithm.

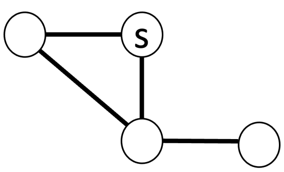
Attribute	Explanation	Initialization
u.status	tells us whether a node is	
u.dist		
u.parent		

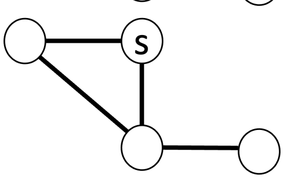
We will also make use of

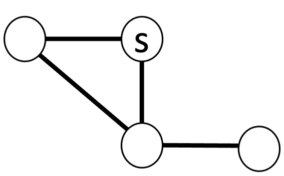
Basic Idea

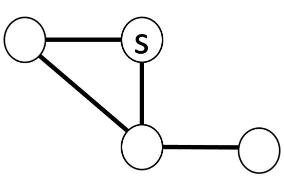
- Mark s as
- Iteratively *explore* discovered nodes
- Continuously update

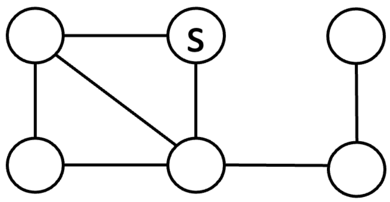
Example

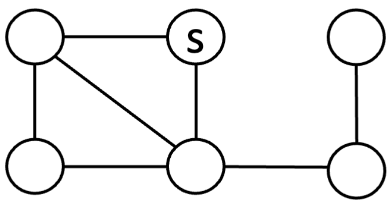


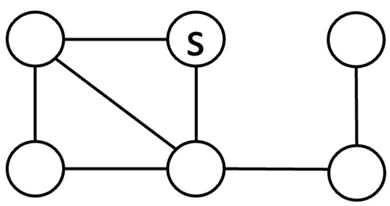


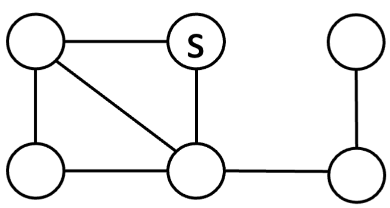


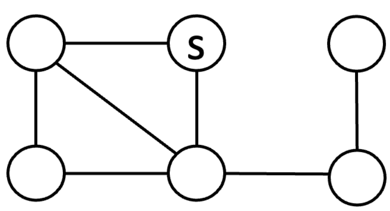


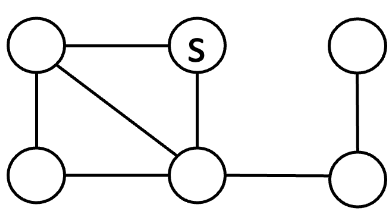

S

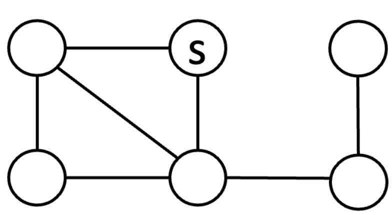

S


S


S


S


S


S

2.1 Shortest Paths and Breadth First Trees

Definition 1. Given a graph $G = (V, E)$, source node s, and a *parent* attribute for each node, a ____________________________ is a subgraph $\hat{G} = (\hat{V}, \hat{E})$ where

It is furthermore a ______________________________ if it contains a unique simple path

from s to v that is ____

Benefits of the BFS algorithm

- If G is undirected, it finds the ____
- It tells us the ____
- It provides a ______________________________

```
BFS(G,s)
  for v ∈ V do
      v.parent = NIL
      v.dist = ∞
      v.status = U
  end for
  s.dist = 0
  s.status = D
  Initialize Q
  Enqueue(s)
  while |Q| > 0 do
      u = Dequeue(Q)
      N(u) = Adj[u]
      for v in N(u) do
          if v.status == U then
              v.status = D
              v.parent = u
              v.dist = u.dist + 1
              Enqueue(v)
          end if
      end for
      u.status = E
  end while
```

2.2 Code and Runtime Analysis

- We assume G is undirected and stored as an adjacency list.
- Initializing attributes takes ____________________
- Each node u only enters Q once, and entering/leaving Q takes __________
- When we *explore* u, we discover up to _________________________

Using aggregate analysis, what is the overall runtime of this method?

3 Depth First Search: Background and Motivating problems

Recall that a *breadth-first* search explores nodes that are k steps away from node s before exploring any nodes that are $k+1$ steps away.

A *depth-first search* instead explores the *most recently discovered vertex* before backtracking and exploring other previously discovered nodes.

Roughly speaking, this is accomplished by __________________________________.

Depth first search is used in several applications for analyzing directed graphs. We will take a closer look at these applications before exploring how to solve them using DFS.

Directed graph reminders

3.1 Reachability and Connected Components

Reachability. Given a graph $G = (V, E)$ and node set $S \subseteq V$, node $v \in S$ is *reachable* from node $u \in S$ if ____.

Connected components. For an undirected graph $G = (V, E)$ a connected component is a maximal subgraph in which every node in is ____

Weakly Connected components If $G = (V, E)$ is directed, a *weakly connected component* is ____

Strongly Connected components If $G = (V, E)$ is directed, a *strongly connected component* is subgraph $S \subseteq V$ in which there is ____

Question 2. *How many weakly connected components and strongly connected components are there in the following graph, respectively?*

A *1 and 3*

B *1 and 2*

C *0 and 1*

D *2 and 3*

3.2 Directed Acyclic Graphs

A *cycle* in a directed graph is a directed path ____

A *Directed acyclic* graph is a directed graph that _______________________.

Examples

3.3 Topological Sorting

A topologically ordering of a directed acyclic graph $G = (V, E)$ is an ordering of nodes so that:

4 Depth First Search Algorithm

Unlike in a BFS, a depth-first search (DFS):

- Explores the *most recently discovered vertex* before backtracking and exploring other previously discovered vertices
- All nodes in the graph are explored (rather than just a DFS for a single node s)
- We keep track of a global *time*, and each node is associated with two timestamps for when it is *discovered* and *explored*.

Each node $u \in V$ is associated with the following attributes

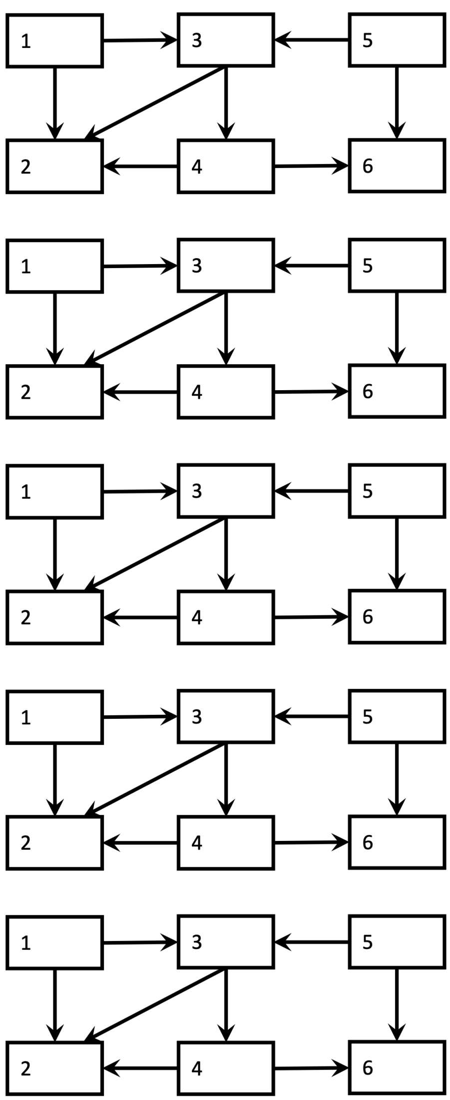
Attribute	Explanation	Initialization
u.status	tells us whether a node has been *undiscovered*, *discovered*, and *explored*	
u.D	timestamp when u is first discovered	
u.F	timestamp when u is finished being explored	
u.parent	predecessor/"discoverer" of u	

DFS(G)

for $v \in V$ **do**
  v.parent $= NIL$
  v.status = U
end for
time = 0
for $u \in V$ **do**
  if u.status $== U$ **then**
    DFS-Visit(G, u)
  end if
end for

DFS-Visit(G, u)

time = time + 1
$u.D$ = time
u.status $= D$
for $v \in$ Adj$[u]$ **do**
  if v.status $== U$ **then**
    v.parent $= u$
    DFS-Visit(G, v)
  end if
end for
u.status = E
time = time + 1
$u.F$ = time



4.1 Runtime Analysis

Question 3. *What is the runtime of a depth first search, assuming that we store the graph in an adjacency list, and assuming that* $|E| = \Omega(|V|)$*?*

- A $O(|V|)$
- B $O(|E|)$
- C $O(|V| \times |E|)$
- D $O(|V|^2)$
- E $O(|E|^2)$

4.2 Properties of DFS

Theorem 4.1. *In any depth-first search of a graph $G = (V, E)$, for any pair of vertices u and v, exactly one of the following conditions holds:*

- $[u.D, u.F]$ *and* $[v.D, v.F]$ *are disjoint;* ____________________________________
- $[v.D, v.F]$ *contains* $[u.D, u.F]$ *and* ____________________________________
- $[u.D, u.F]$ *contains* $[v.D, v.F]$ *and* ____________________________________

4.3 Classification of Edges

Given a graph $G = (V, E)$ performing a DFS on G produces a graph $\hat{G} = (V, \hat{E})$ where

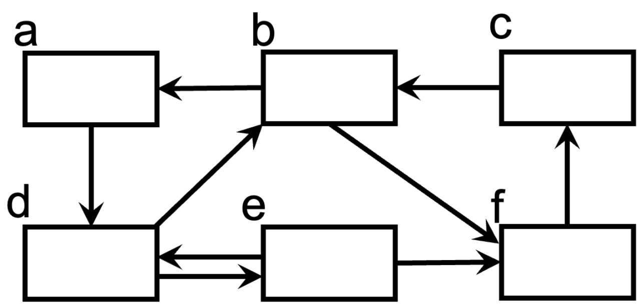
$$\hat{E} = \{(u.\text{parent}, u) \colon v \in V \text{ and } v.\text{parent} \neq NIL\}$$

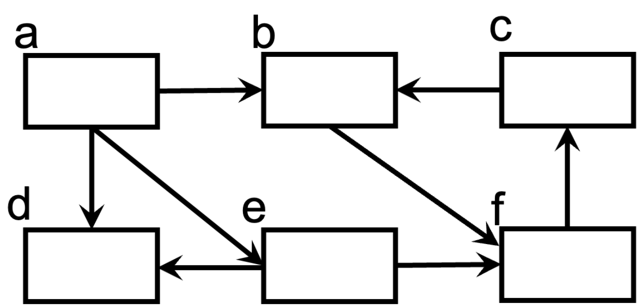
This is called a *depth-first* forest of G.

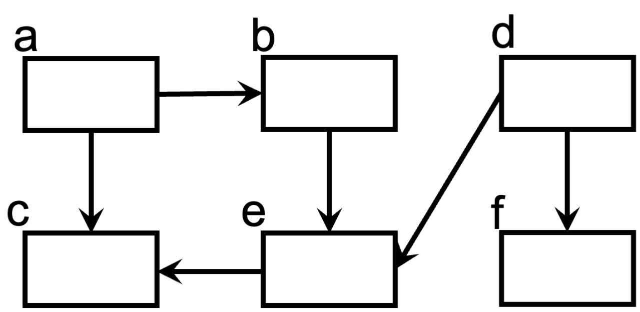
Given any edge $(u, v) \in E$, we can classify it based on the status of node v when we are performing the DFS:

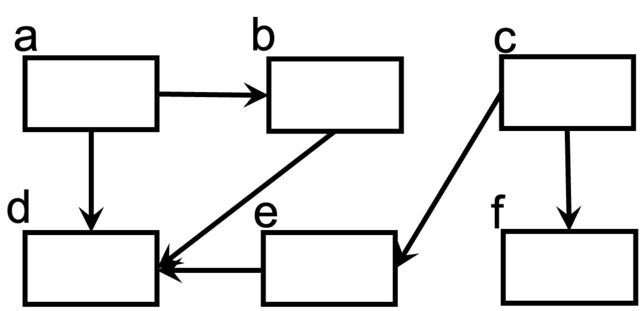
Edge	Explanation	How to tell when exploring (u, v)?
Tree edge	edge in $\hat{E}$	
Back edge	connects u to ancestor v	
Forward edge	connects vertex u to descendant v	*and* $u.D < v.D$
Cross edge	either (a) connects two different trees or (b) crosses between siblings/cousins in same tree	*and* $u.D > v.D$

5 Practice









Question 4. *How many of the above graphs were directed acyclic graphs?*

A *1*

B *2*

C *3*

D *4*

E *none of them*