CS1800

11/3 - Fri :)

Admin

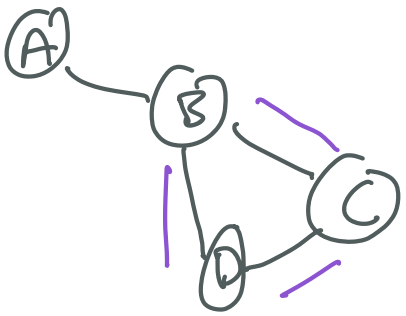
- Hw5 due 11:59pm
- Hw6 out, due next Fri
- exam #2 in two weeks :)
  - 11/17 (Fri)
  - 2 hour window (1 hr 40 min exam)
  - on gradescope
  - DRC accommodation
  - review in recitation that week

Agenda - graphs!

1. Breadth First Search
2. Depth First Search
3. Dijkstra's Algorithm

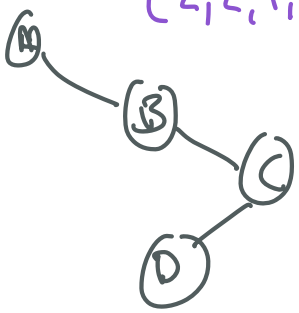
} graph algorithms

0. Review of graphs



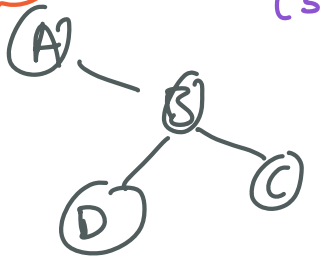
- undirected
- unweighted
- cyclic
- strongly connected
- simple

deg sequence (2,2,1,1)

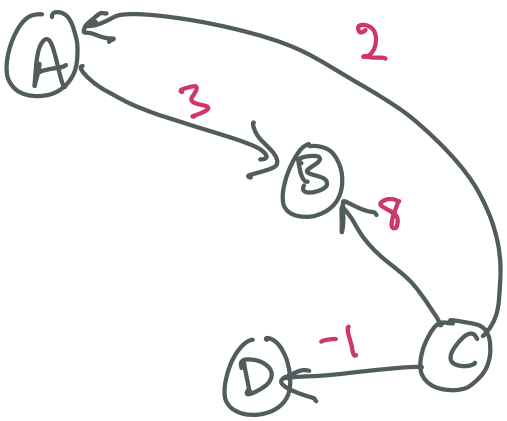


→ not isomorphic! ←

deg seq (3,1,1,1)



- trees!



- directed
- weighted
- not connected
- not simple

Adj list

A : B-3

B :

C : A-2, B-8, D--1

D :

Adj matrix

	A	B	C	D
A	0	3	0	0
B	0	0	0	0
C	2	8	0	-1
D	0	0	0	0

1. Breadth First Search (BFS)

- searching a graph
- **not** looking for specific vertex or edge
- **instead** find everything reachable from starting point

(construct path from start to every reachable vertex)

- finding directions on a map

↓

how expensive is path?

- weighted: sum of weights
- unweighted: # edges

BFS Search #1

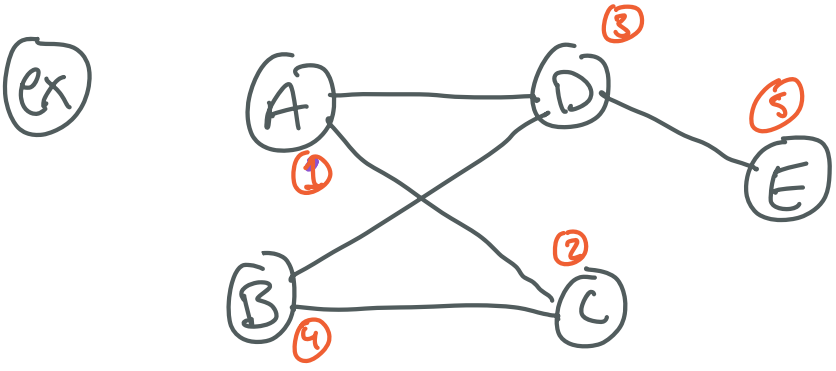
- works best on unweighted
- find a path from source to every reachable vertex } always
- find the shortest path } unweighted

Keep track of:

1. order in which we discover vertices
2. predecessor in path

Algorithm:

- start at source vertex
- discover all vertices reachable in one step
- discover all vertices reachable in two steps
- ... (in case of tie, go alphabetize)



visit A's neighbors - 1 step

2 steps - reachable from C, D

BFS start at A

Order we discover

A, C, D, B, E

one step | two steps

Predecessors

A	B	C	D	E
/	C	A	A	D

How do I get from...

A to C? A, C cost = 1

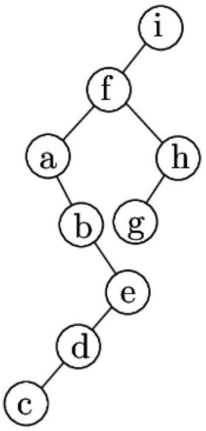
A to B? A, C, B cost = 2

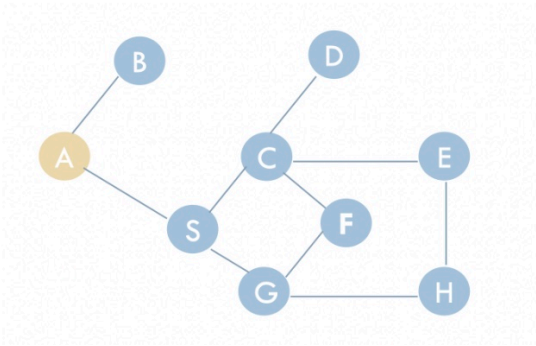
A to E? A, D, E cost = 2

order in which we discover

BFS start at i

BFS starting at A





i, f, a, h, b, g, e, d, c

1 step, 2 steps, 3 steps, 4, 5, 6

Path from i to b?

i, f, a, b cost = 3

A, B, S, C, G, D, E, F, H

1 step, 2, 3 steps

Path from A to E?

A, S, C, E cost = 3

10:50

2. Depth First Search (DFS)

- start at a given vertex
- find all reachable vertices

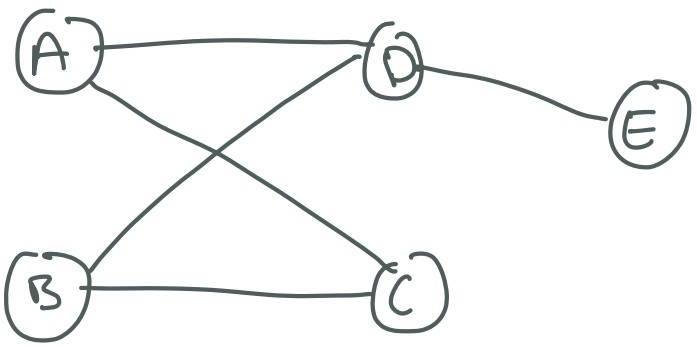
Track of...

1. order in which we discover vertices
2. predecessors on path

BFS (unweighted) guarantees shortest path

DFS guarantees a path from source to every reachable vertex

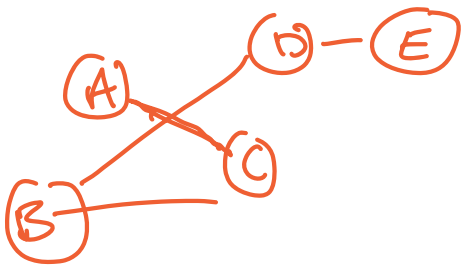
(ex) go down a single path
until we need to backtrack
we backtrack as little as possible!



(break ties in alpha order)

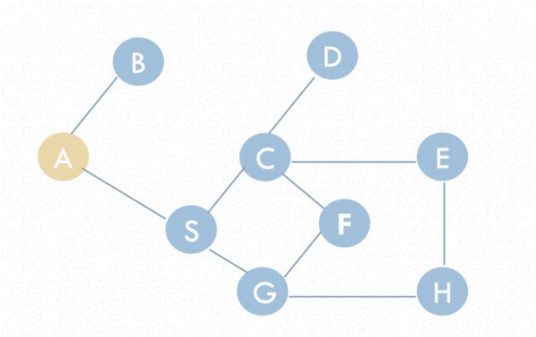
A, C, B, D, E

result: tree

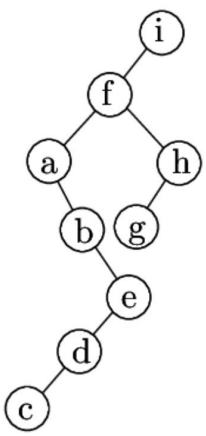


order in which we discover vertices?

DFS start at A



DFS start at i



A, B, S, C, D, E, H, G, F

i, f, a, b, e, d, c, h, g.

Path from A to E?

A, S, C, E cost = 3

Predecessors:	A	B	S	C	D	E	F	G	H
	/	A	A	S	C	C	G	H	E

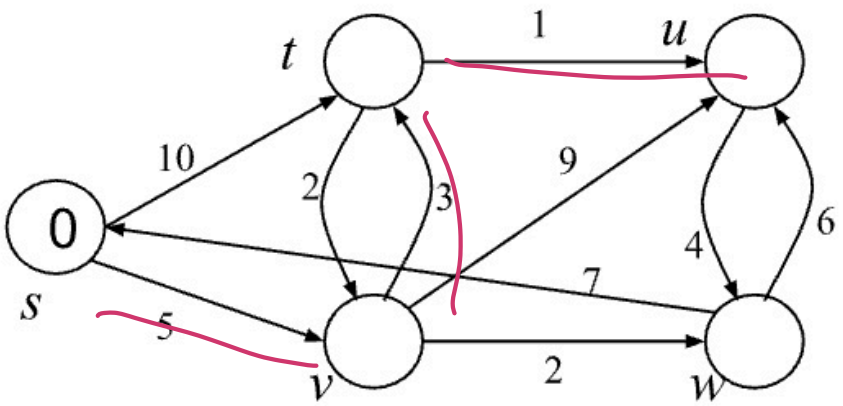
Dijkstra's Algorithm

- Start a given vertex
- weighted graph (non-neg edge weights)
- find best path from source to every reachable vertex

Try something, replace with better if we find one

- assign value to every vertex
  - ↳ weight from source
  - ↳ start ∞ except source
- always pick smallest to go next
- replace value, if smaller, with current edge/path

Dijkstra



values

S	T	U	V	W
0	∞	∞	∞	∞

min

S

replace?

T = 10 S, T

V = 5 S, V

T U V W

10 ∞ 5 ∞

V

T = 8 S, V, T

U = 14 S, V, U

W = 7 S, V, W

T U W

8 14 7

W

S no change

U = 13 S, V, W, U

T U

8 13

T

U = 9 S, V, T, U

V no change

U

W no change

U

9

Shortest path to...

S	/	0
T	S, V, T	8
U	S, V, T, U	9
W	S, V, W	7
V	S, V	5