

Graphs!

# About me

John Wright

4<sup>th</sup> year at Berkeley

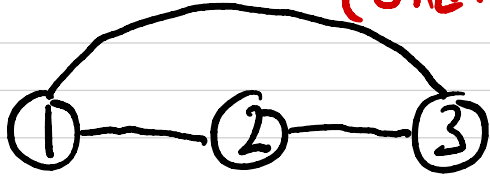
Office: Soda 683

Office hours: Wednesdays 11am - noon

\*this week only\*: Offes Thursday 9am - 10am

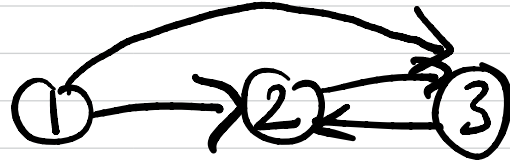
# Graphs

(undirected)



$$G = (V, E)$$

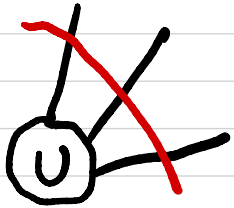
(directed)



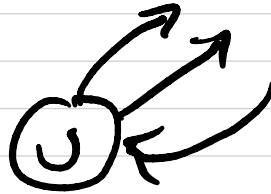
$$(u, v) \in E \text{ if } u \rightarrow v$$

Parameters:  $n = |V|$   
 $m = |E|$

$$(m \leq n^2)$$



$$\text{deg}(u) = 3$$



$$\begin{aligned} \text{in-deg}(u) &= 1 \\ \text{out-deg}(u) &= 2 \\ &\text{aka } \text{deg}(u) \end{aligned}$$



Facebook friend graph  
(undirected)

- $n = 2.91$  billion users
- avg user has 338 friends
- $m = 500$  billion edges

Question: Who are my friends?



Best 6 min 15 min 6 min

Yifang Taiwan Fruit Tea, 2516 Bancroft Way

Soda Hall, Berkeley, CA 94709

Add destination

Options

Send directions to your phone Copy link

via Sather Rd 15 min 0.6 mile  
Details

via Eshleman Rd 15 min 0.6 mile

via Barrow Ln 14 min 0.6 mile

All routes are mostly flat

Search along the route

Restaurants Coffee Groceries Things to do

Yifang Taiwan Fruit Tea

Soda Hall

15 min 0.6 miles

14 min 0.6 miles

15 min 0.6 miles

University of California, Berkeley

Sather Gate

Memorial Glade

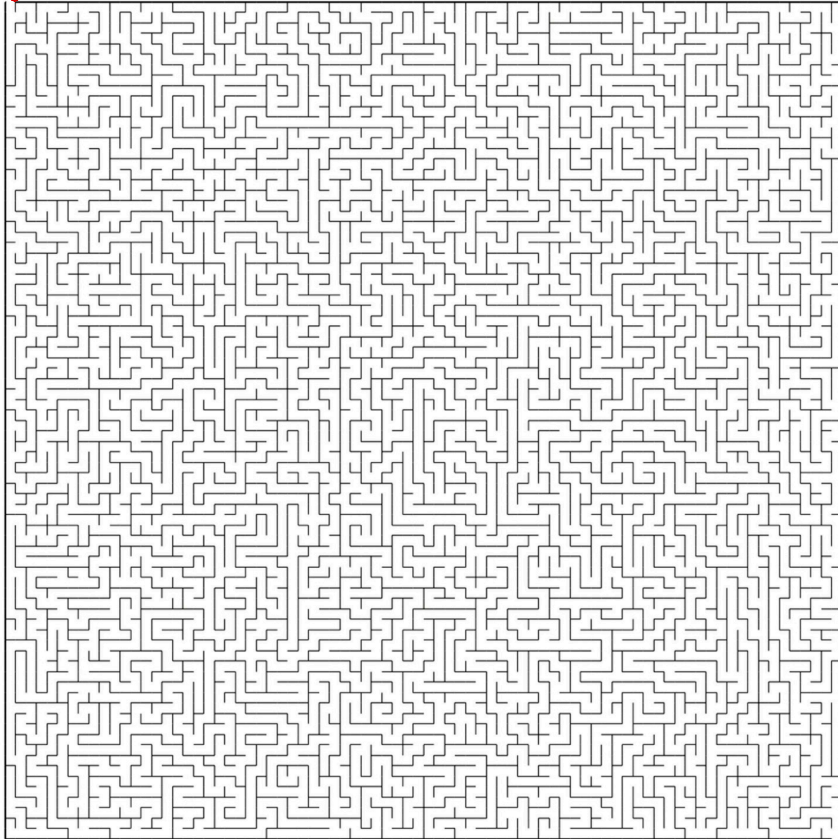
Hearst Memorial Mining Building

Yifang Taiwan Fruit Tea

Google Maps

Q: Shortest path to boba?

entrance  $u$



exit  $v$

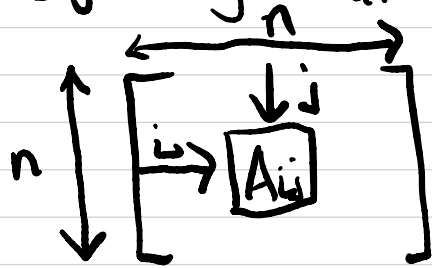
Maze solving

Q: Is  $u$  connected  
to  $v$ ?

# Representing graphs on computers

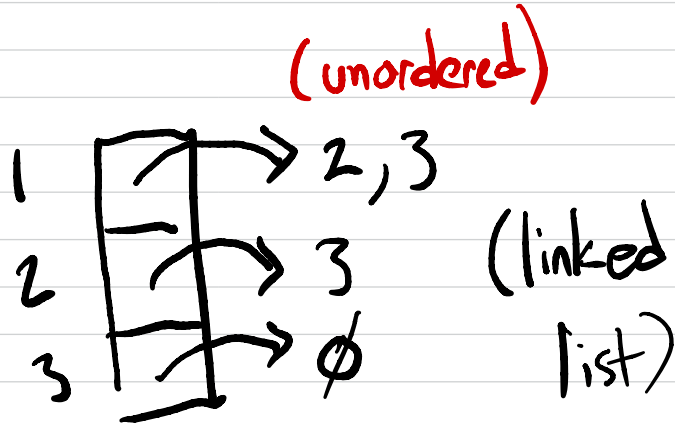
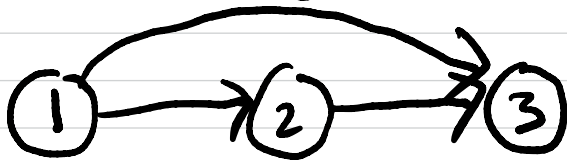
$$V = \{1, \dots, n\}$$

(1) adjacency matrix representation



$$A_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{o.w.} \end{cases}$$

(2) adjacency list representation



	Adjacency matrix	Adjacency lists
size	$O(n^2)$	$O(n+m)$
time to: {	answer is $(u,v) \in E$	$O(n)$ or $O(\deg(u))$
	enumerate $u$ 's neighbors	$O(\deg(u))$

Facebook graph w/ adjacency matrix

size  $n^2 = (2.91 \text{ billion})^2$  space

$= 8.5 \times 10^{18} = 1 \text{ million TB}$

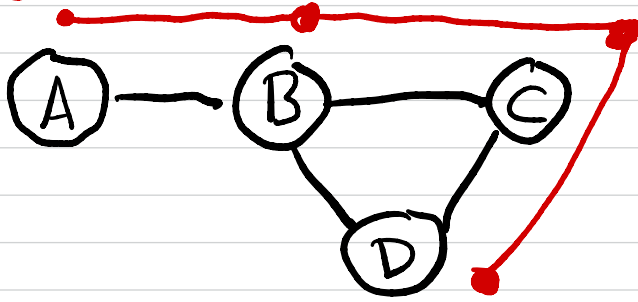
computing friends list = 2.91 billion steps

# Graph Exploration

Useful for:

1. Is there a path from  $u$  to  $v$ ?
2. Is  $G$  connected?
3. What are  $G$ 's connected components?

Connectivity for undirected graphs ( $\exists$  path from  $u$  to  $v$ ?)



**explore(G, u)**  
visited[u] = true

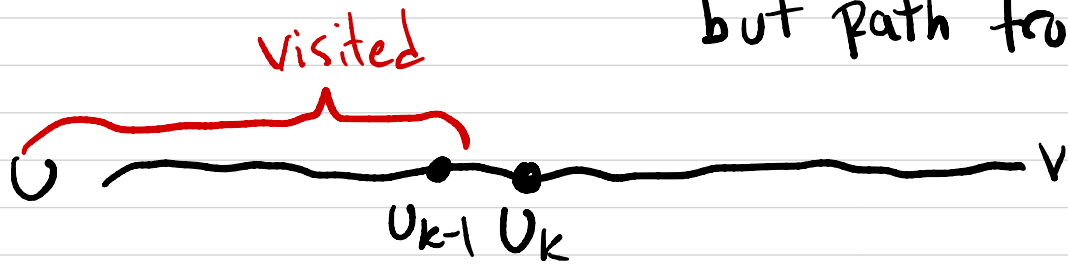
for  $v$  s.t.  $(u, v) \in E$   
if visited[v] = false  
explore(G, v)

boolean array visited [n]  
(init to all 0's)

**Property:**  $\text{explore}(G, u)$  visits exactly the vertices  $v$   
s.t.  $G$  has a path from  $u$  to  $v$

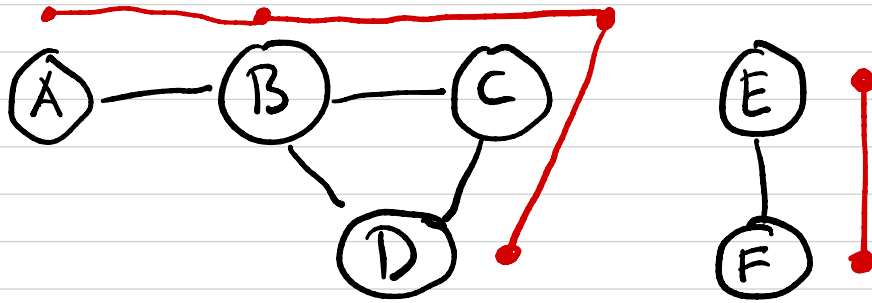
Pf: 1.  $v$  explored  $\Rightarrow$  path from  $u$  to  $v$   
2. path from  $u$  to  $v \Rightarrow$  explore  $v$

Assume false, so  $v$  not explored,  
but path from  $u$  to  $v$



Should call  $\text{explore}(G, u_k)$ . Contradiction!

# Connectivity for undirected graphs ( $\exists$ path from $u$ to $v$ ?)



$explore(G, u)$   
 $visited[u] = true$

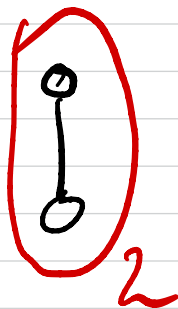
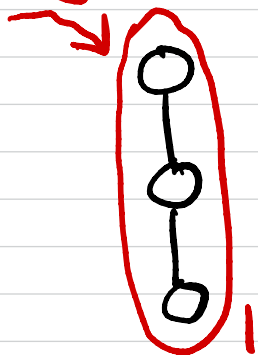
for  $v$  s.t.  $(u, v) \in E$   
if  $visited[v] = false$   
 $explore(G, v)$

$dfs(G)$   
boolean array  $visited[n]$   
(init to all 0's)

for  $v \in V$   
if  $visited[v] = false$   
 $explore(G, v)$



Computing  $G$ 's connected components (undirected)



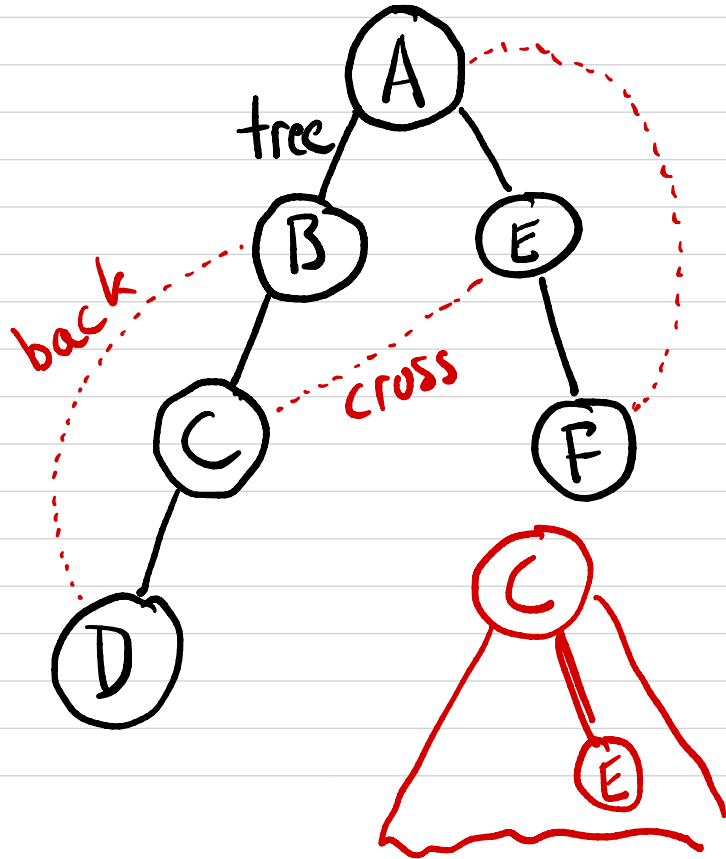
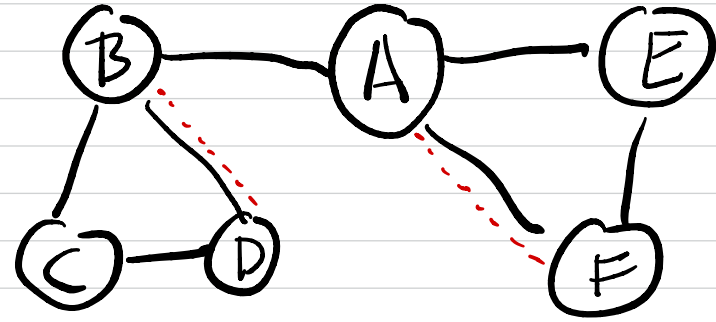
(3 connected components)

```
explore(G, u)
  visited[u] = true
  ccnum[u] = count
```

```
for v s.t. (u, v) ∈ E
  if visited[v] = false
    explore(G, v)
```

```
dfs(G)
  boolean array visited[n]
  (init to all 0's)
  count = 1
  int array ccnum[n]
  for v ∈ V
    if visited[v] = false
      explore(G, v)
      count = count + 1
```

# The DFS tree



# Runtime of DFS

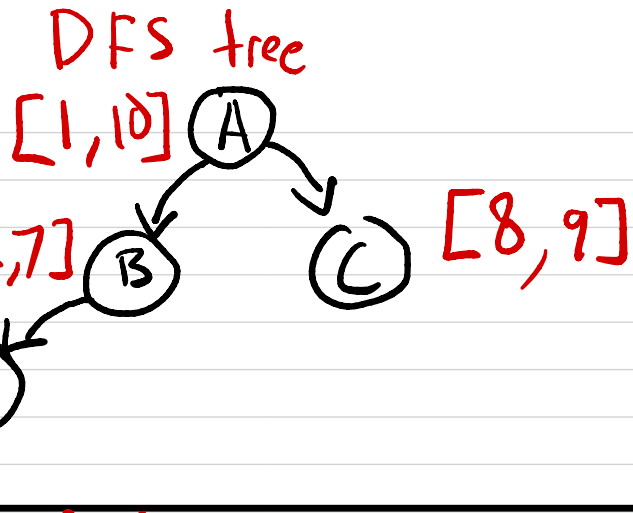
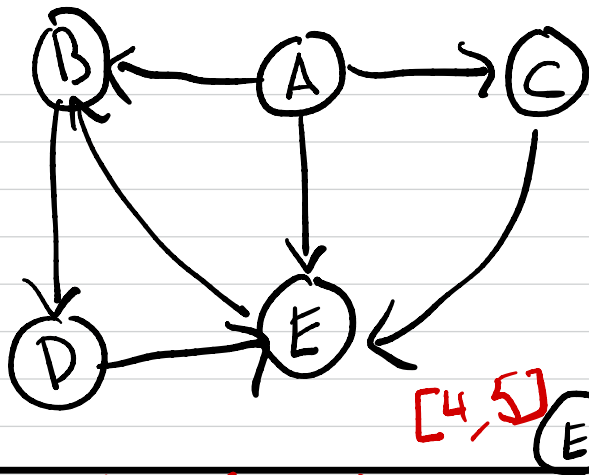
Only call `explore(G, u)` exactly once, for each  $u$

Runtime of `explore(G, u)`

- set `visited[u] = true`  $O(1)$  time
- enumerate  $u$ 's neighbors  
 $O(\text{deg}(u))$  time (linked list)

$$\text{total} = \sum_{u \in V} O(1 + \text{deg}(u))$$

$$\Rightarrow O(n + m) \text{ time}$$



explore (G, u)

visited [u] = true

pre[u] = clock

clock = clock + 1

for v s.t. (u, v) ∈ E  
 if visited [v] = false  
 explore (G, v)

post[u] = clock  
 clock = clock + 1

dfs (G)

boolean array visited [n]  
 (init to all 0's)

clock = 1

int array pre [n], post [n]

for v ∈ V  
 if visited [v] = false  
 explore (G, v)