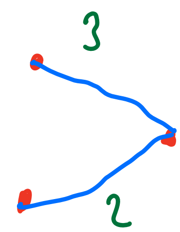
CS 331, Fall 2026 Today: Graphs

Mini-lecture 4 (9/9)

Graphs

Collection of vertices V

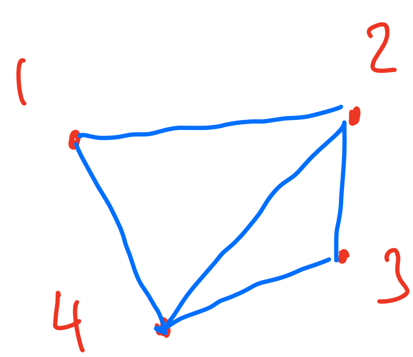
Connected by edges $E \subseteq V \times V$



... possibly with edge weights $w \in \mathbb{R}^E$

In this class, n = # vertices
m = # edges } reserved letters

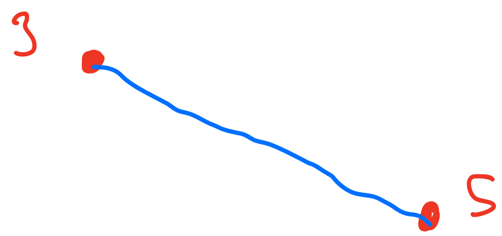
We often arbitrarily identify $V \equiv [n]$

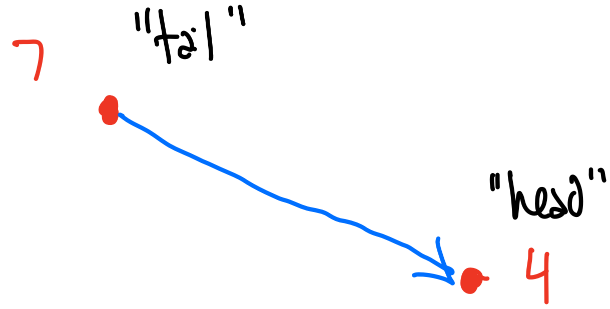


$n = 4$
$m = 5$

Basic types of graphs:

Directed / Undirected





Undirected:

edges = unordered tuples (sets)

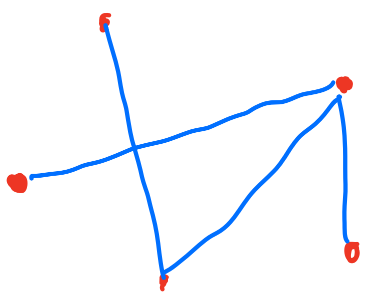
e.g. $e = \{3, 5\}$

directed:

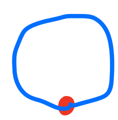
edges = ordered tuples

e.g. $e = (7, 4)$

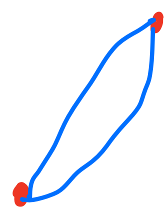
Simple / Multigraph



simple = no



self loops



parallel edges

Ex. How large can m be as function of n

when graph is ...simple? ...multigraph?

Subgraph

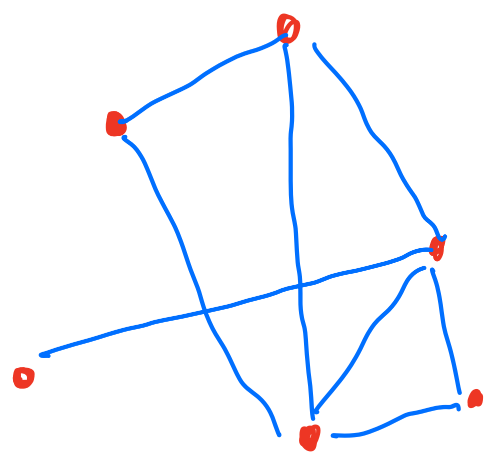
H is subgraph of G if

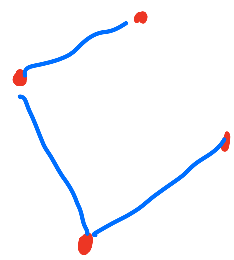
- delete some vertices
- delete some edges

We call it <u>induced</u> if

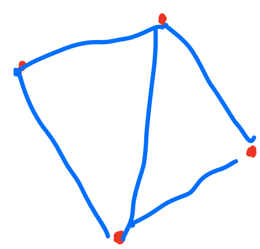
- only delete vertices
- keep all edges on those vertices

G



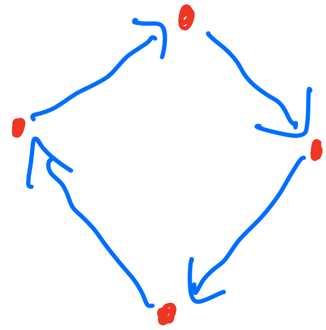


Subgraph H



induced subgraph I

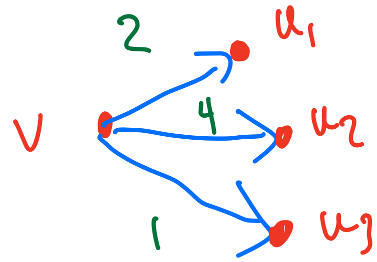
Paths / Cycles



Can be undirected / directed

Ex. In n-vertex path/cycle, how big is m?

$$\underset{\text{degree}}{\deg(v)} = \sum_{e=(v,u)\in E} w_e$$



$$= \left|\{u : (v,u) \in E\}\right|$$

(out-) neighbors in unweighted graph

Theorem: In unweighted, undirected graph,

let $O := \{v \in V : \deg(v) \text{ is odd}\}$

Then, O has even size.

Proof: Count $\sum_v \deg(v)$ in two ways

① $$\sum_{e=(u,v)} \underset{\deg(u)}{1} + \underset{\deg(v)}{1} = 2|E| \quad \text{(even)}$$

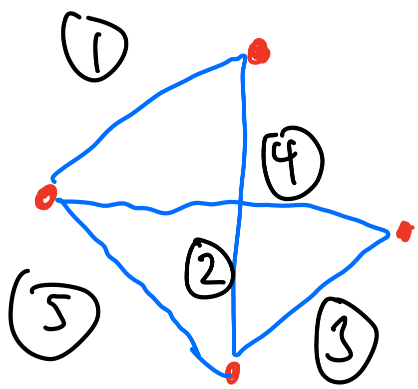
② $$\sum_{v\in O} (\text{odd } \#) + \sum_{v \notin O} (\text{even } \#)$$

must be even # of $v \in O$

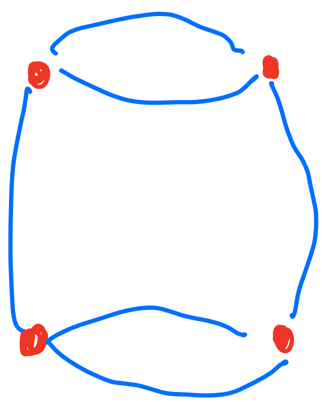
"Application"

An Euler tour (every edge used once, "pen never leaves paper")

<u>all</u> vertices have even degree except possibly start & end



proof: if you <u>enter</u> you also <u>leave</u> } pairs

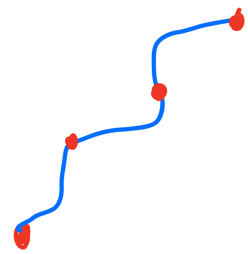


all degrees odd $\Rightarrow$ ✗

Trees

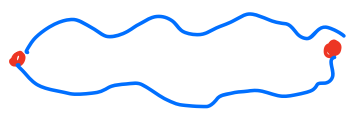
Forest = <u>undirected</u> graph w/ no <u>cycles</u>

Tree = <u>connected</u> forest

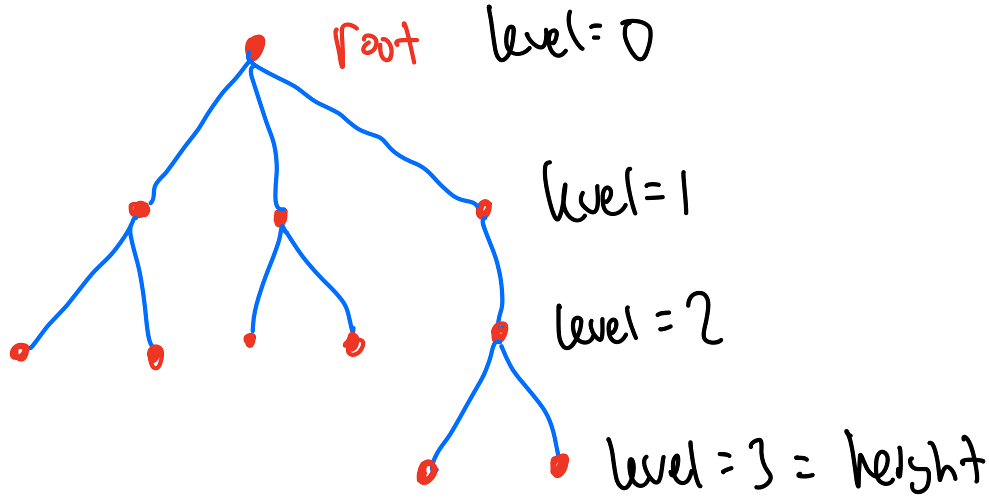


∃ path between any vertices

Theorem: For every u, v vertices in tree there's a <u>unique</u> path between u & v

Proof: If two, that's a cycle!  $\blacksquare$

Convenient orientation system: rooted tree



level = length of <u>unique</u> root-to-leaf path

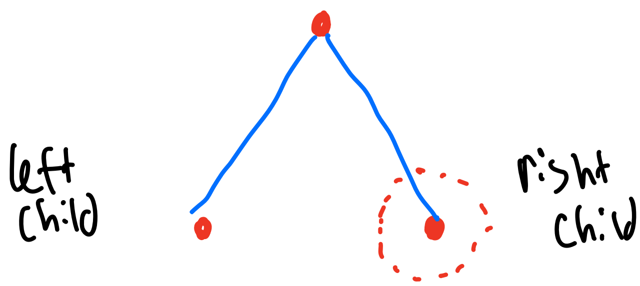
Binary trees

Rooted tree, every

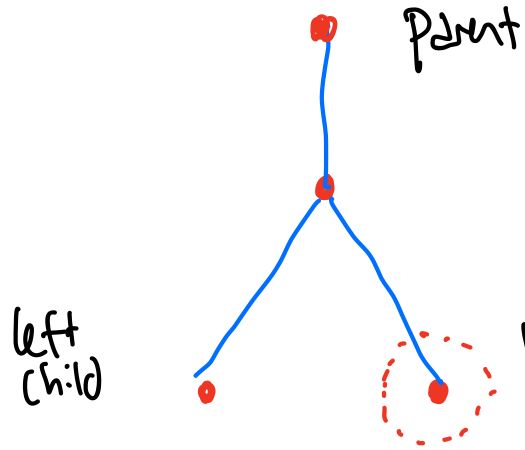
$\deg(v) \in \{1, \underbrace{2, 3}_{\text{internal vertex}}\}$

1: leaf

Root

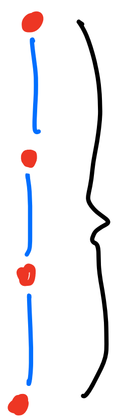


Internal vertex

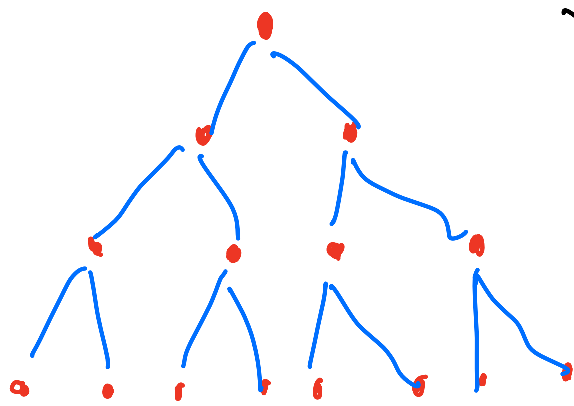


(recursively, ancestor)

Two extremes: height h



$h+1$ vertices



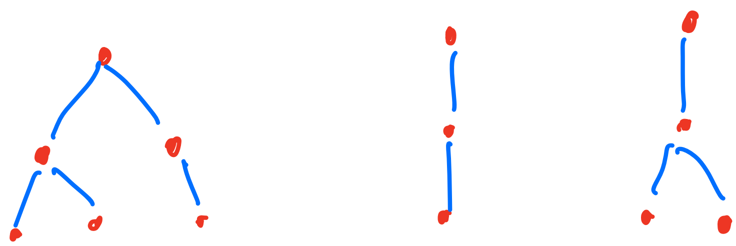
Complete binary tree

$1+2+\dots+2^h = 2^{h+1}-1$ vertices

Theorem: Forest w/ k connected components
n vertices
has $n-k$ edges. (Tree $= n-1$)

Proof: Split into 3 steps.

① Enough to prove for trees

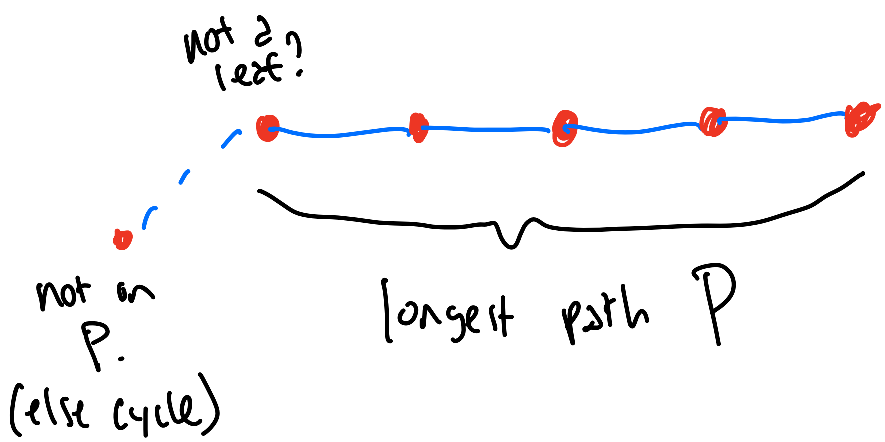


$k = 3$

\# edges $\quad n_1 - 1 \quad n_2 - 1 \quad n_3 - 1 \quad = n - k$

② Claim: both endpoints of <u>longest path</u> in a tree are leaves. (So every tree has ≥ 2 leaves)

Suppose otherwise, contradict longest path.



(3) Induction on n: every tree has $m = n - 1$

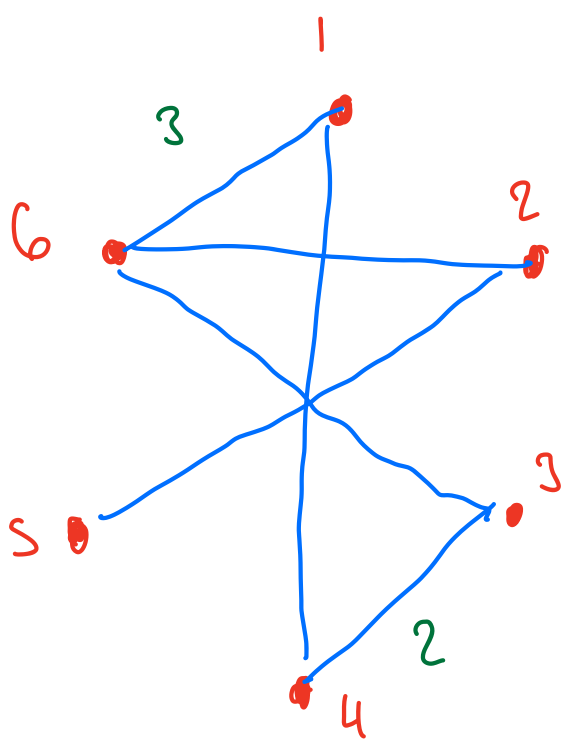
Clearly true if $n = 1$ (base case)

Otherwise, delete leaf + edge $\Rightarrow$ $n-1$ vertices, $n-2$ edges $\blacksquare$

Representing a graph

Warmup: adjacency matrix

	1	2	3	4	5	6
1				1		3
2					1	1
3				2		1
4	1		2			
5		1				
6	3	1	1			



Problem: Takes n^2 time to initialize,
n time to look up neighbors

In this class: <u>adjacency list</u> model $(L_V^{in}, L_V^{out}, L_E)$

L_V^{out}: vertices know edges

L_E: edges know vertices

vertices		
1	(1,2)	
2	(2,4)	(2,5)
3	(3,5)	
4	(4,2)	
5		

(pointers to L_E)

edges	out	in	weights
(1,2)	1	2	$w_{(1,2)}$
(2,4)	2	4	$w_{(2,4)}$
(2,5)	2	5	$w_{(2,5)}$
(3,5)	3	5	$w_{(3,5)}$
(4,2)	4	2	$w_{(4,2)}$

pointers to L_V^{in}, L_V^{out}

Space: $O(n + m)$

Primitives:

- "pass over all edges"

  runtime: $O(m)$

- "pass over neighbors of v"

  runtime: $O(\deg(v))$