

CSSE 230 Day 28

Graphs and their representations

After this lesson, you should be able to ...
... explain what makes a graph different than a tree
... implement simple graph algorithms

<https://www.google.com/maps/preview#!data=!1m4!1m3!1d989355!2d-87.4496039!3d38.8342589!4m2!3m1!7!1m5!1sRose-Hulman+Institute+of+Technology%2C+5500+Wabash+Ave%2C+Terre+Haute%2C+IN+47803!2s0x886d6e421b703737%3A0x96447680305ae1a4!3m2!3d39.482156!4d-87.322345!1m1!1sHoliday+World+%26+Splashin'+Safari%2C+Santa+Claus%2C+IN!3m8!1m3!1d245622!2d-86.923997!3d39.325645!3m2!1i1920!2i955!4f13.1!5m2!13m1!1e1!7m4!1!1m3!1m1!1e1!2b1&fid=0>

Graphs

Terminology

Representations

Algorithms

Example Graph

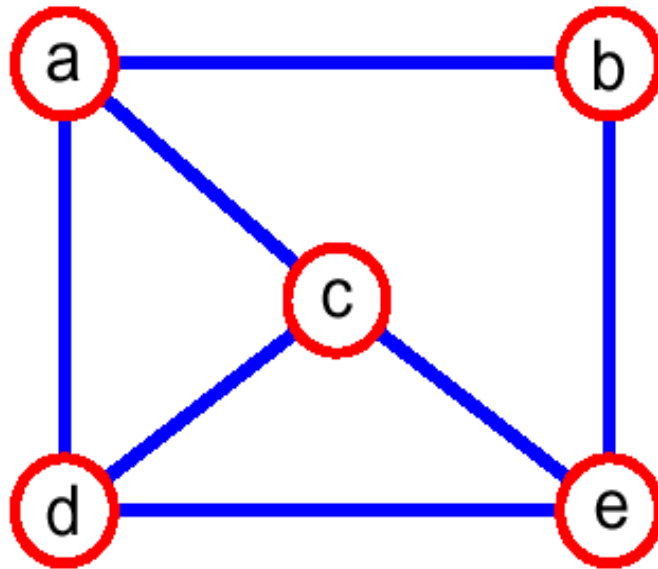
A graph $G = (\mathbf{V}, \mathbf{E})$ is composed of:

\mathbf{V} : set of *vertices*

\mathbf{E} : set of *edges* connecting the *vertices* in \mathbf{V}

An **edge** $e = (u,v)$ is a pair of **vertices**

Example:



$\mathbf{V} = \{a, b, c, d, e\}$

$\mathbf{E} =$

$\{(a,b), (a,c), (a,d),$
 $(b,e), (c,d), (c,e),$
 $(d,e)\}$

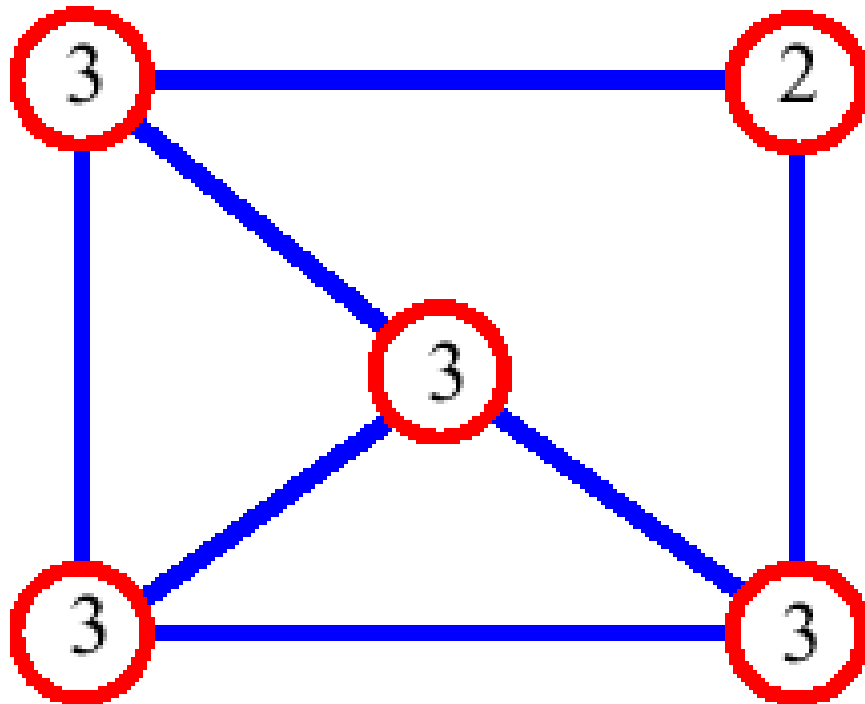
Graph Terminology

also called
“neighbors”

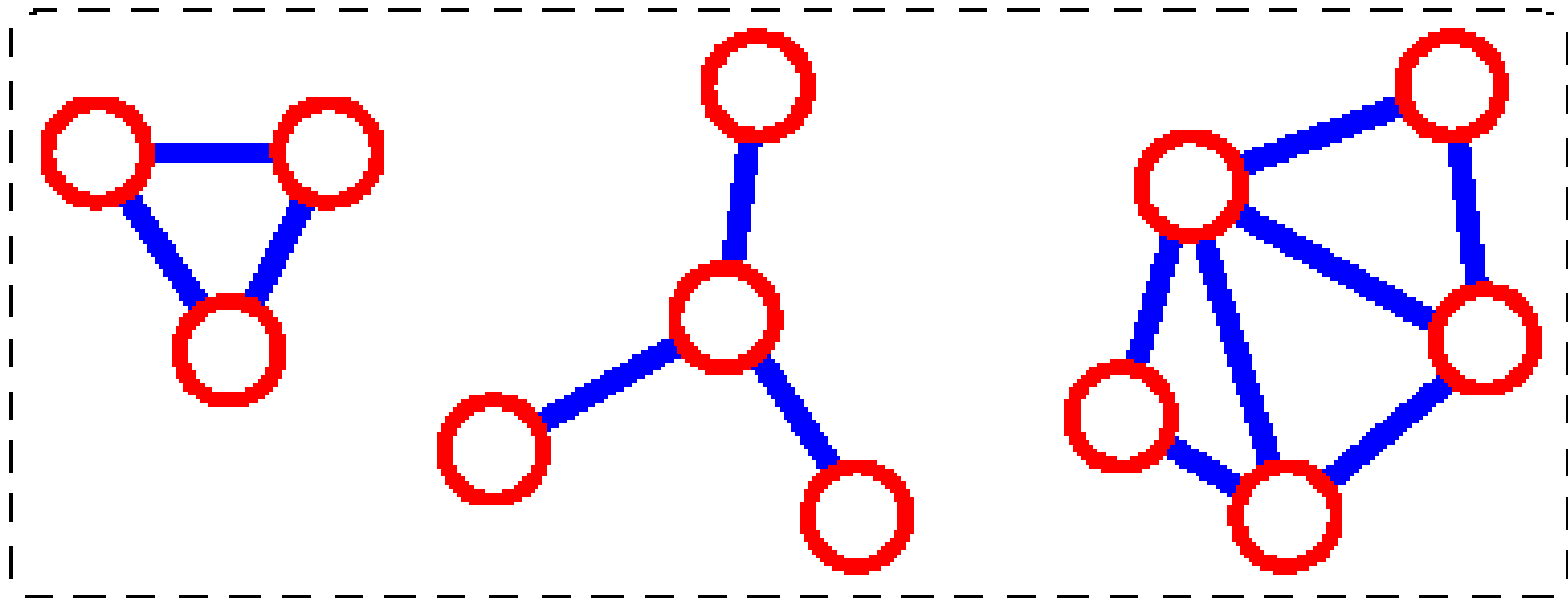
- **adjacent vertices**: connected by an edge
- **degree** (of a **vertex**): # of adjacent vertices

$$\sum_{v \in V} \deg(v) = 2(\# \text{ edges})$$

- Since adjacent vertices each count the adjoining edge, it will be counted twice



connected component: maximal connected subgraph. E.g., the graph below has 3 connected components.



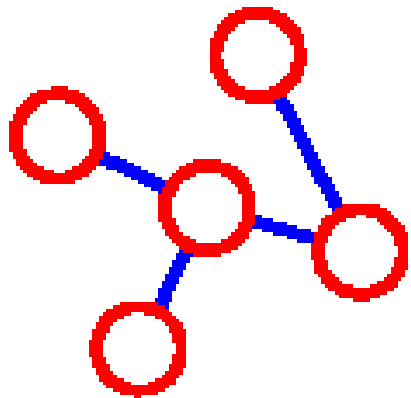
More Connectivity

n = #vertices

m = #edges

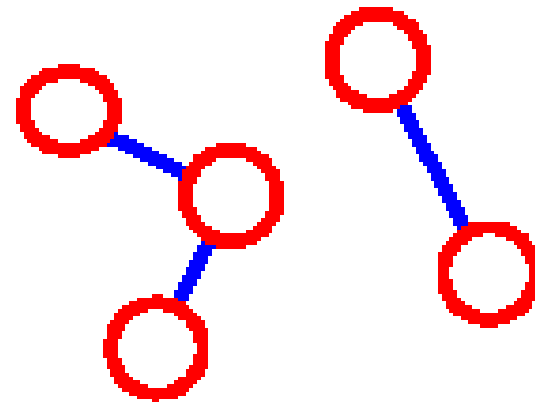
For a tree **m** = **n** - 1

A necessary but not sufficient condition for a graph to be a tree.



$$\mathbf{n} = 5$$
$$\mathbf{m} = 4$$

If **m** < **n** - 1, G is not connected



$$\mathbf{n} = 5$$
$$\mathbf{m} = 3$$

We represent vertices using a collection of objects

- ▶ Each Vertex object contains information about itself
- ▶ Examples:
 - City name
 - IP address
 - People in a social network

There are many options for representing edges of a graph

- ▶ Adjacency matrix
- ▶ Adjacency list. Each vertex stores...
 - pointers to other vertices?
 - named vertices using a `HashMap<Name,Vertex>`
 - An index into an array of the Vertex objects In each case, we need a way to store the vertex collection
- ▶ Edge list

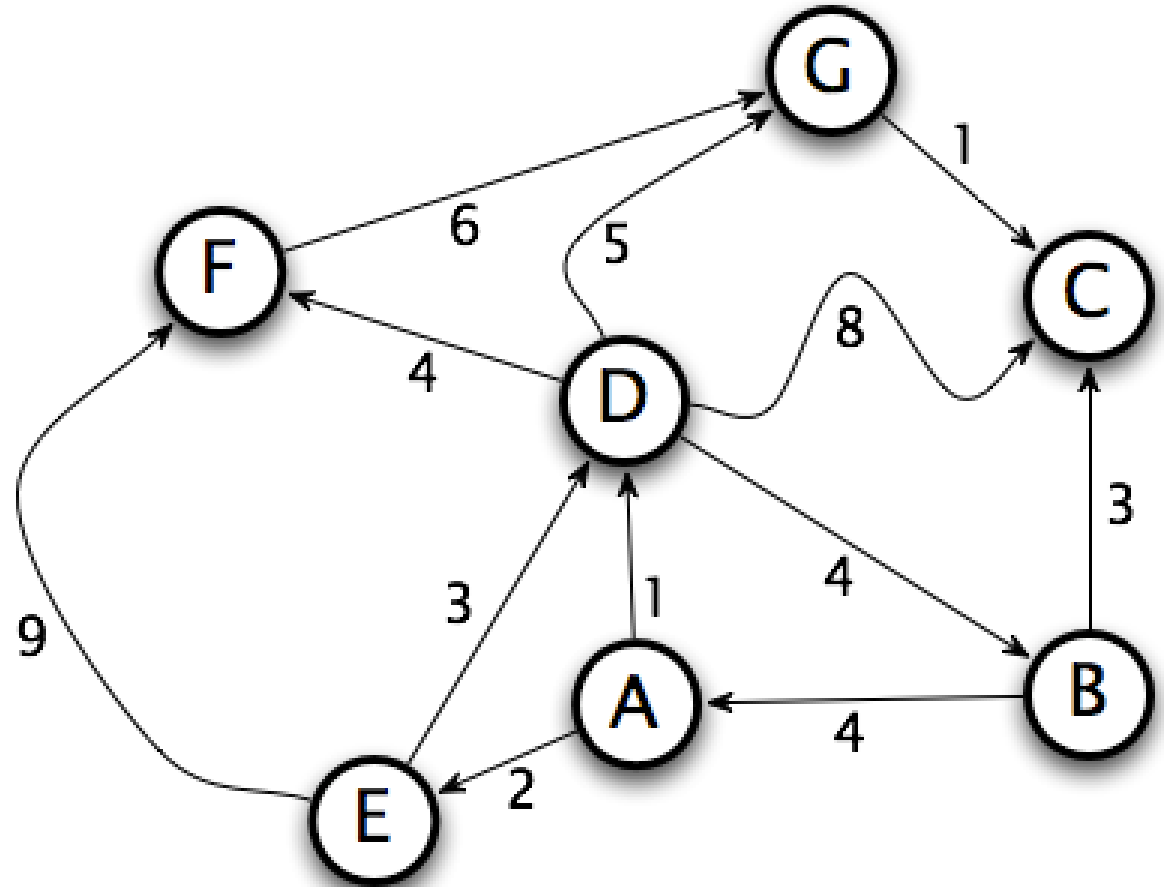
To consider:

Why not just use a triangular “matrix”?

Does a boolean adjacency matrix make sense?

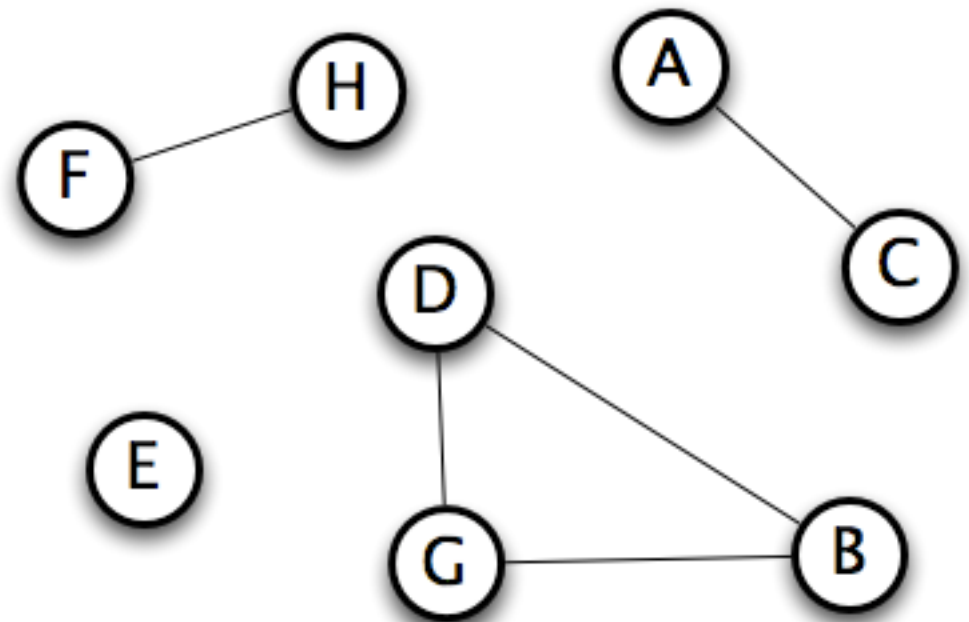
Sample graph problem: Weighted Shortest Path

- ▶ What's the cost of the shortest path from A to each of the other nodes in the graph?



For much more on graphs, take MA/CSSE 473 or MA 477

- ▶ What's the size of the largest connected component?



Check out from SVN:
RandomGraphs