IV

GRAPHS AND TREES

INTRODUCTION TO GRAPHS (KR 10.1)

- Introduction
- Examples of graph models

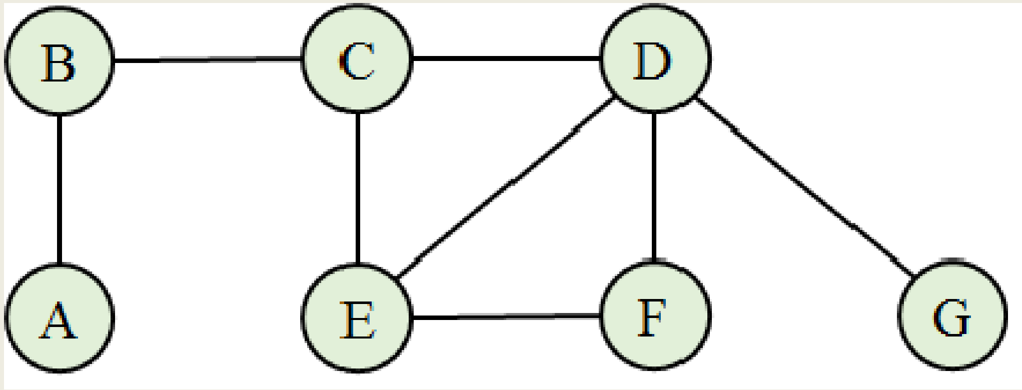
Graphs

Definition: A *graph G* = (*V, E*) consists of a nonempty set *V* of *vertices* (or *nodes*) and a set *E* of *edges.*

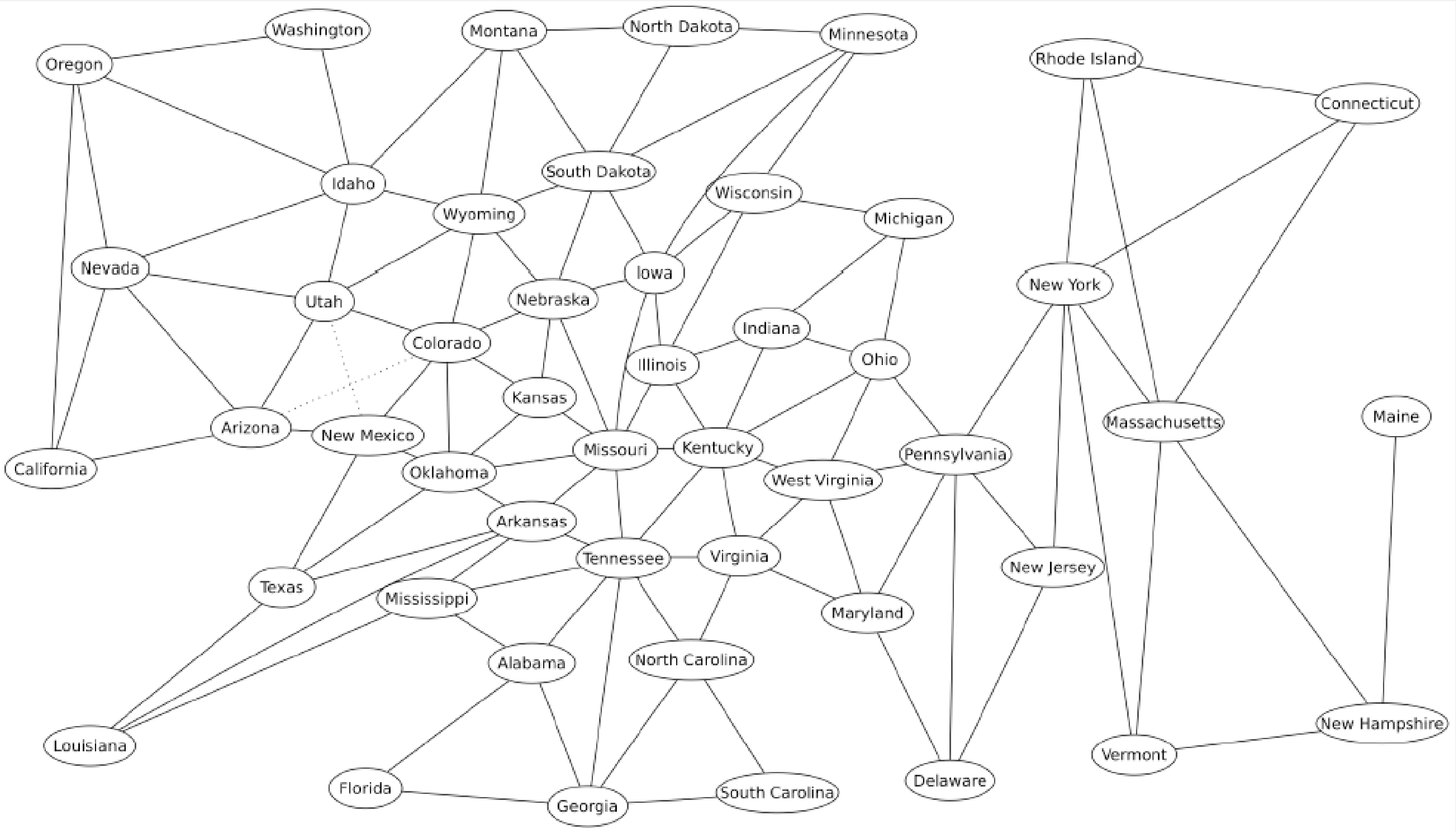
Each edge has either one or two vertices associated with it, called its *endpoints*.

V = {A,B,C,D,E,F,G}

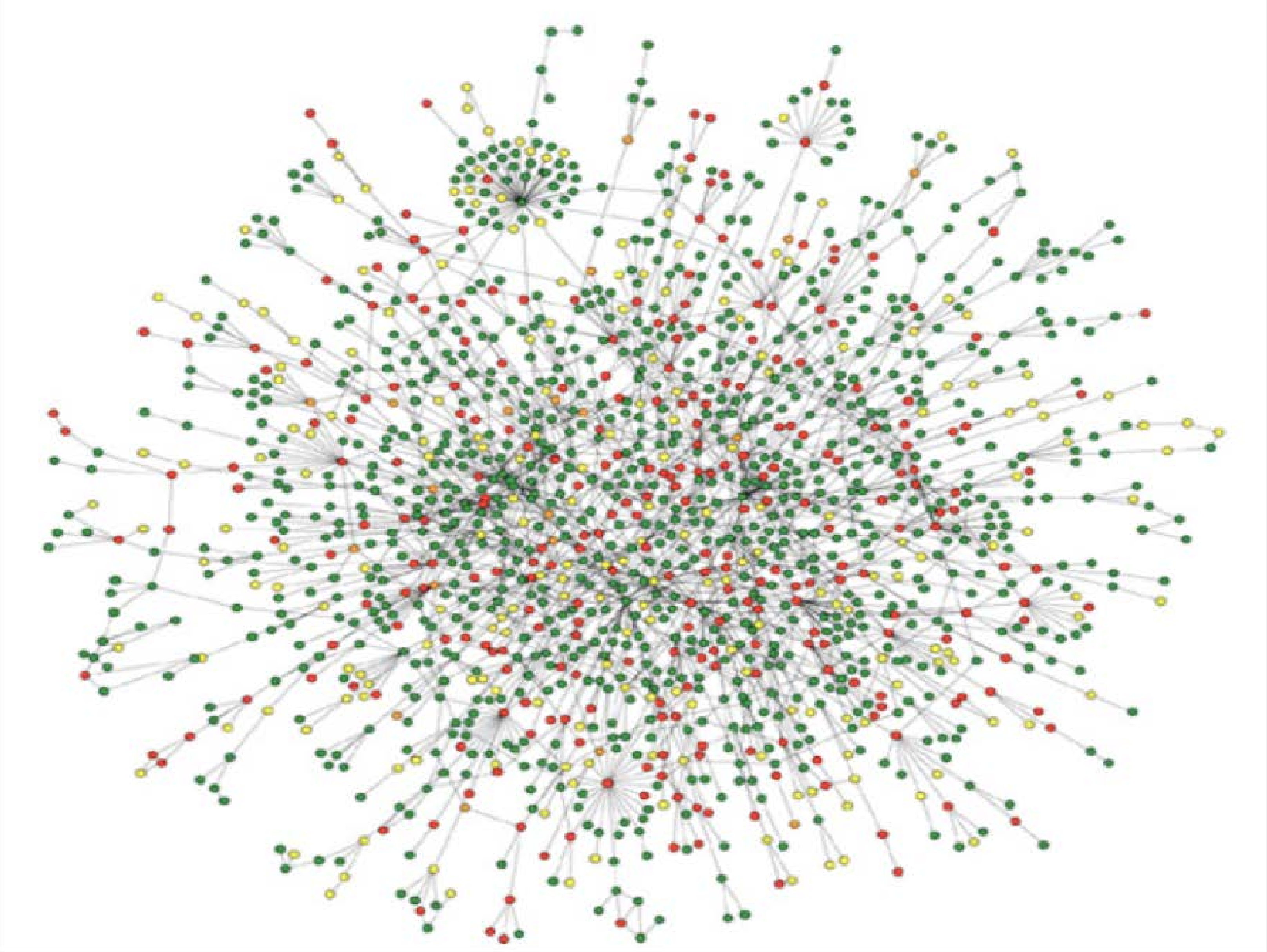
E = {(A,B), (B,C), (C,D), (C,E), (D,E), (D,F), (E,F), (D,G)}



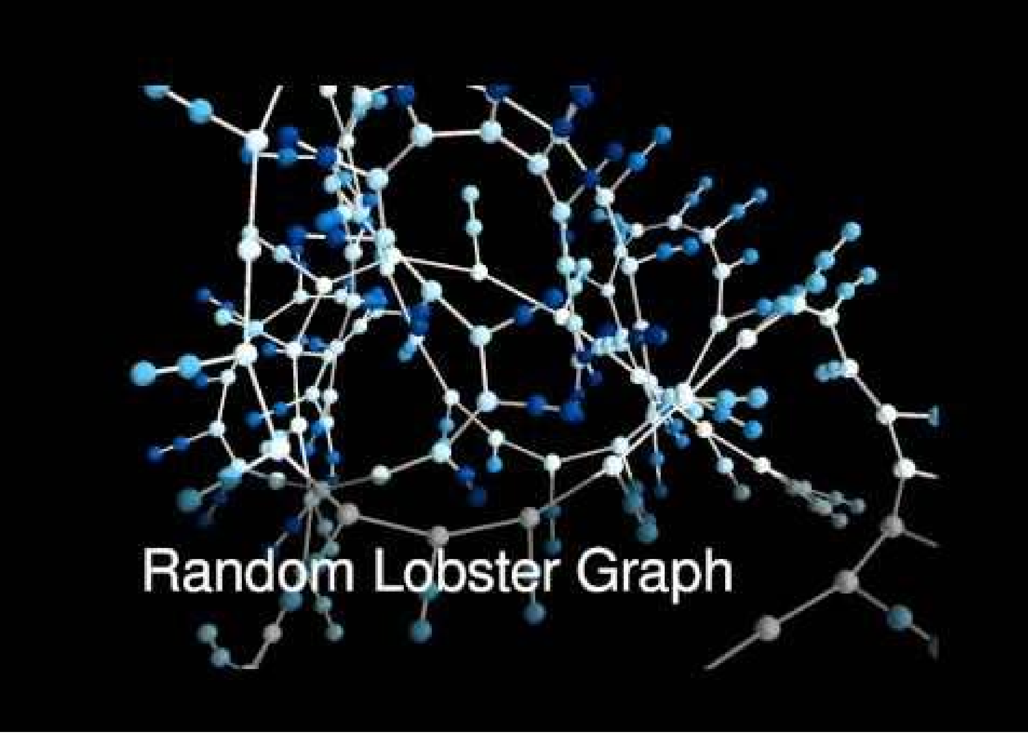
Border graph of 48 contiguous United States

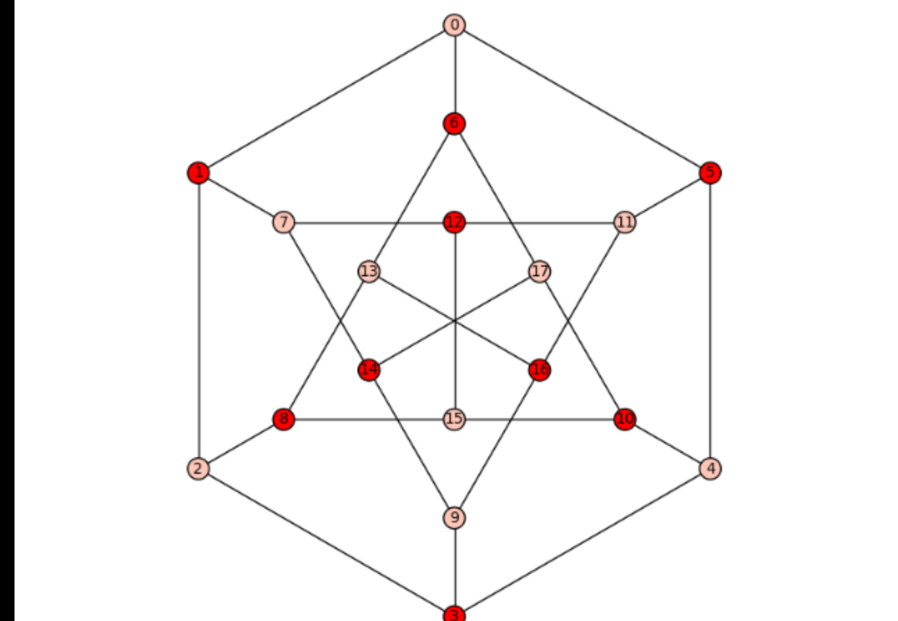


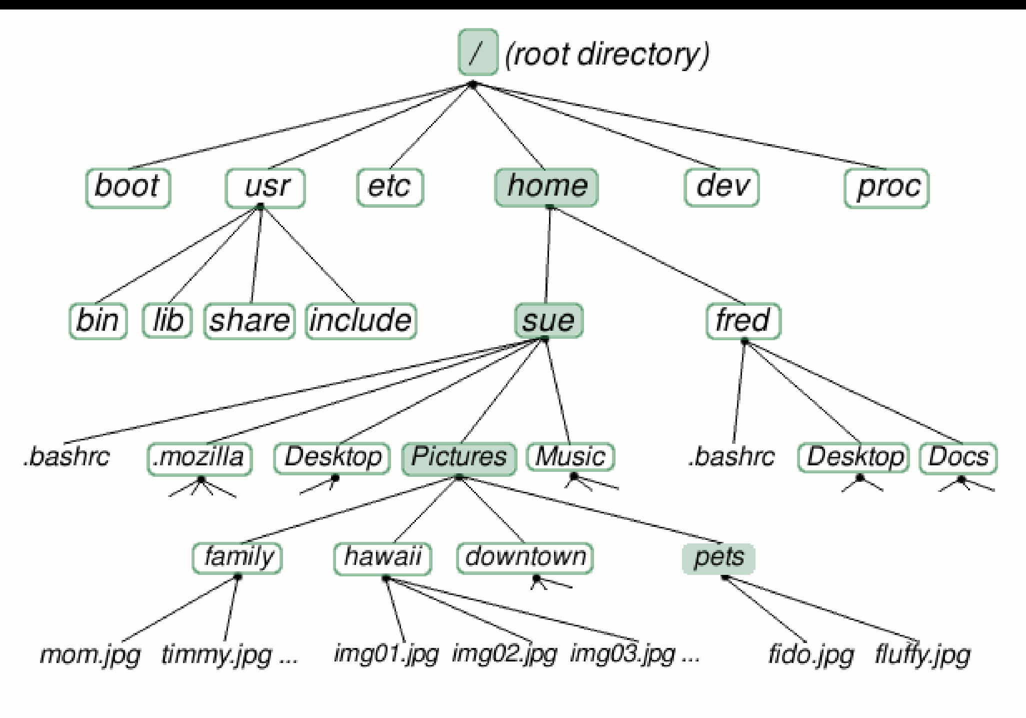
Protein-protein interaction network

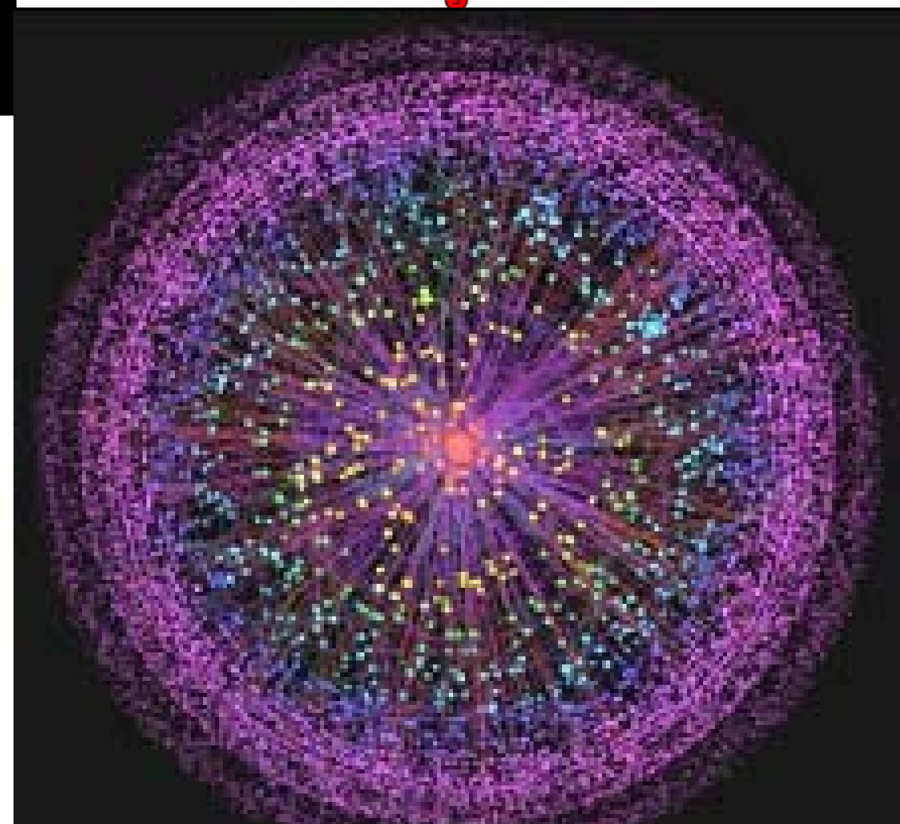


Reference: Jeong et al. Nature Review | Genetics


Random Lobster Graph


0
6
1
5
7
12
11
13
17
14
16
8
15
10
2
4
9
3


/ (root directory)
boot
usr
etc
home
dev
proc
bin
lib
share
include
sue
fred
.bashrc
.mozilla
Desktop
Pictures
Music
.bashrc
Desktop
Docs
family
hawaii
downtown
pets
mom.jpg
timmy.jpg ...
img01.jpg
img02.jpg
img03.jpg ...
fido.jpg
fluffy.jpg

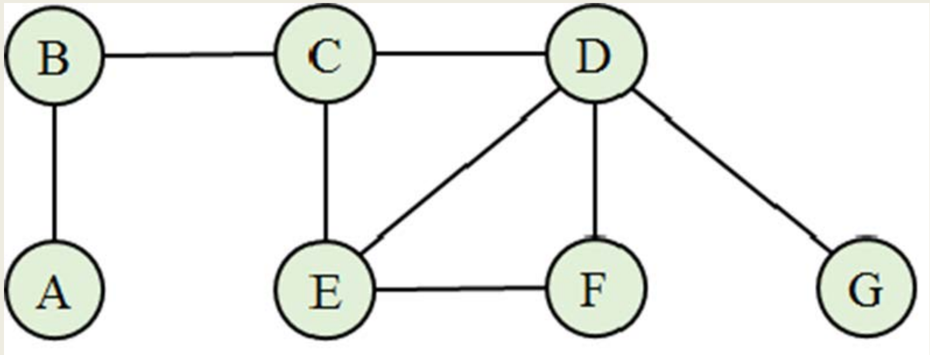


Graph applications

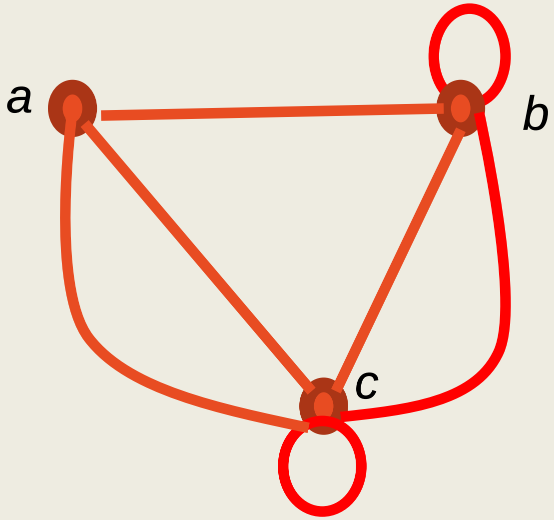
graph	vertex	edge
communication	telephone, computer	fiber optic cable
circuit	gate, register, processor	wire
mechanical	joint	rod, beam, spring
financial	stock, currency	transactions
transportation	intersection	street
internet	class C network	connection
game	board position	legal move
social relationship	person	friendship
neural network	neuron	synapse
protein network	protein	protein-protein interaction
molecule	atom	bond

Some graph terminology

- *Simple graph:* each edge connects two different vertices.
- *Multigraphs* may have multiple edges connecting the same two vertices.
- An edge that connects a vertex to itself is called a (self) *loop*.
- A *pseudograph* may include loops, as well as multiple edges connecting the same pair of vertices.



Example of a pseudograph with multiple edges and loops.



Note: There is no standard terminology for graph theory. It is crucial to understand the terminology being used whenever reading material about graphs.

Directed Graphs

Definition: A *directed graph* (or *digraph*) $G = (V, E)$ consists of a nonempty set V of *vertices* (or *nodes*) and a set E of *directed edges* (or *arcs*).

Each edge is associated with an ordered pair of vertices.

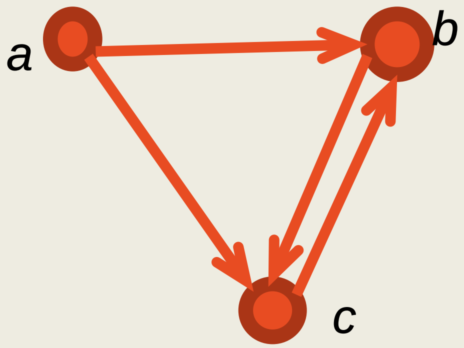
The directed edge associated with the ordered pair (u,v) is said to *start at u* and *end at v*.

Remark: in *undirected graphs edges* have no direction (an edge is associated with an unordered pair)

More Terminology

A *simple directed graph* has no loops and no multiple edges.

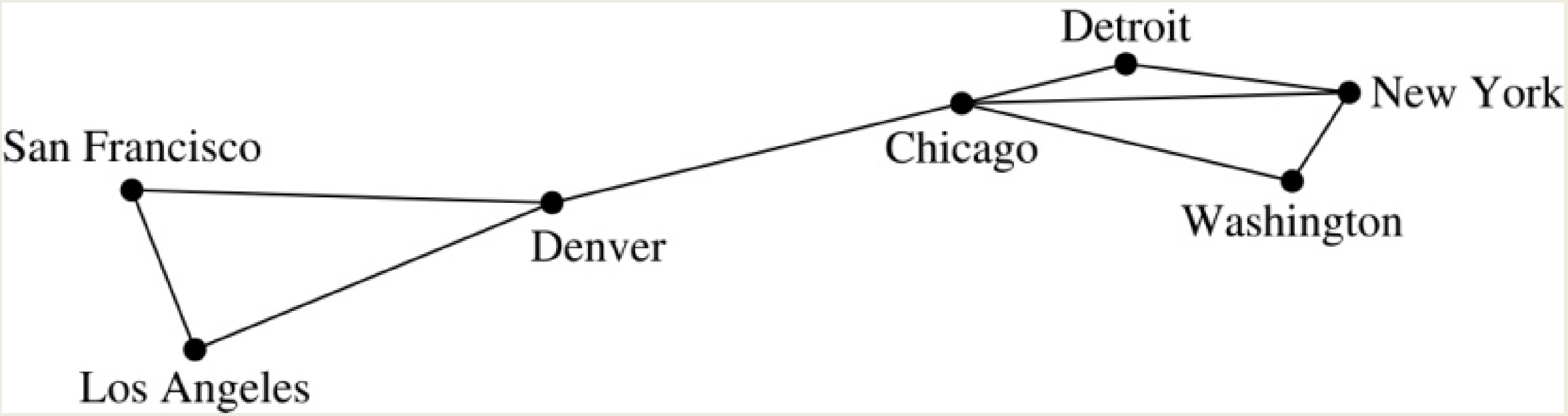
A directed graph with three vertices and four edges, (a,b), (b,c), (c,b) and (a,c)



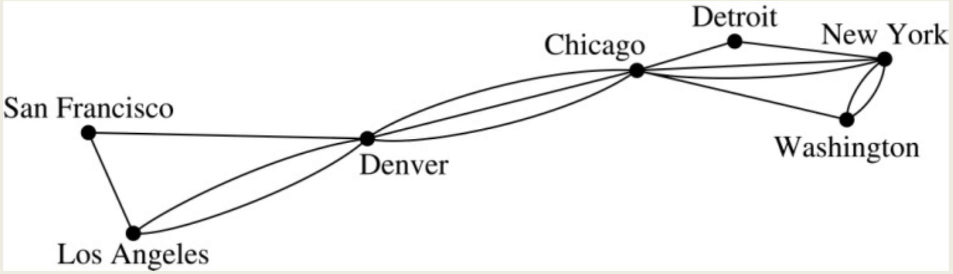
Graph Models: Computer Networks

When building a graph model, use the appropriate type of graph to capture the relevant features of the application.

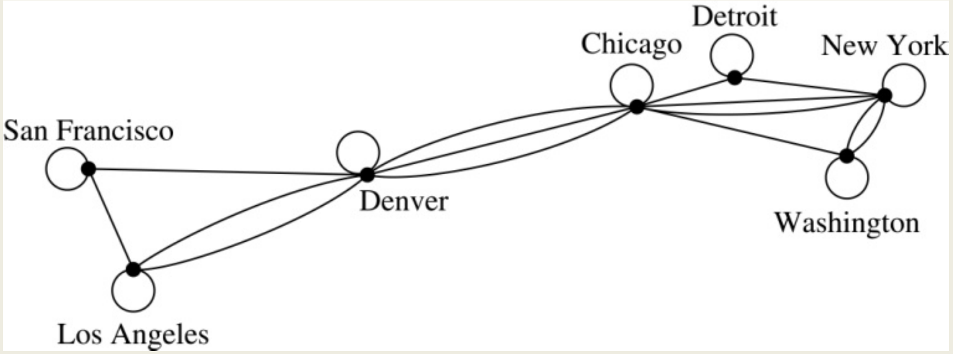
- In a computer network, vertices can represent data centers and the edges represent communication links.



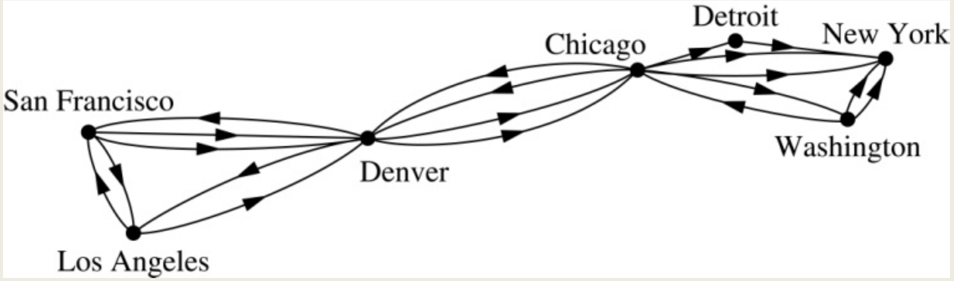
To model a computer network where we care about the number of links between data centers, use a multigraph.



To model diagnostic links at data centers, introduce loops



To model a network with multiple one-way links, use a directed multigraph.



How to build the most effective graph model?

To understand the structure of a graph when building a graph model, consider:

- Are the edges of the graph undirected or directed (or both)?
- Are multiple edges needed?
- Are self-loops present?
- What entries (e.g., weights) should be associated with edges or vertices?

Examples of Applications of Graphs

Social networks
- Friends graphs, collaboration graphs
- Influence graphs

Communications networks
- Call graphs

Information networks
- Web graphs
- Citations graphs

Software design
- Module dependency graphs

Transportation networks
- Airline routes graphs
- Road networks graphs

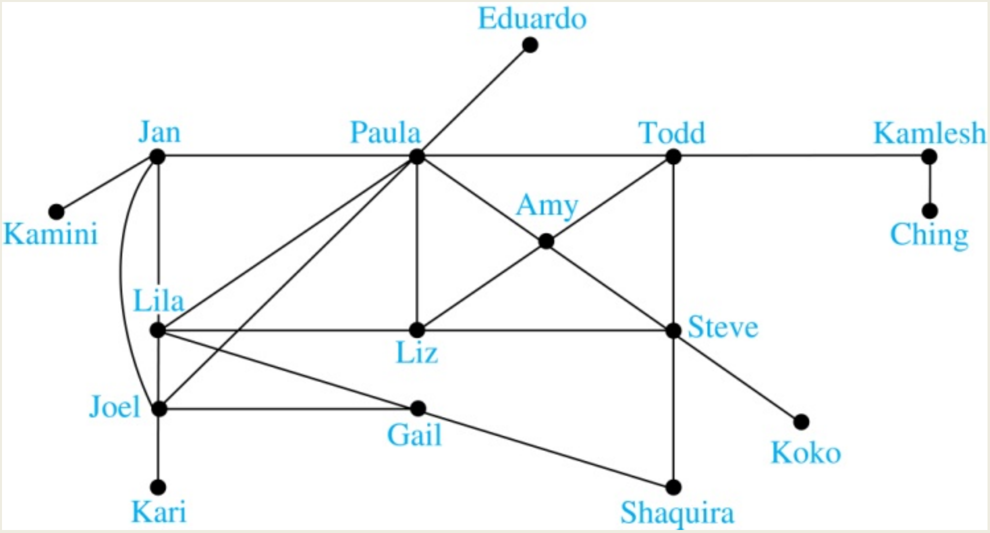
Graph Models: Social Networks

In a *social network*, vertices can represent individuals or organizations and edges represent relationships between them.

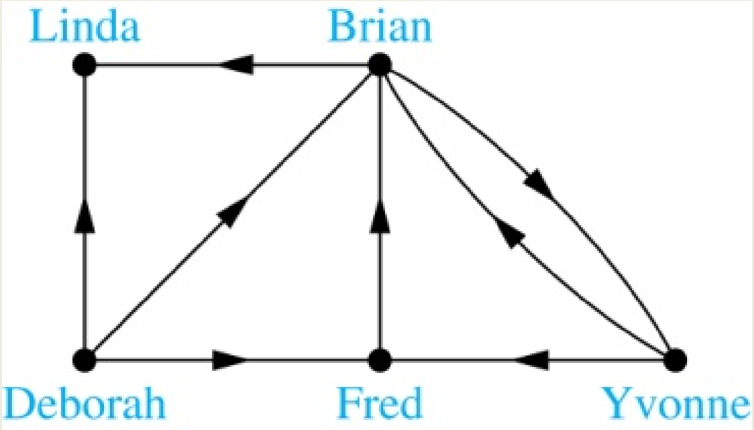
Useful graph models of social networks include:

- ***friendship graphs*** - undirected graphs where two people are connected if they are friends.
- ***collaboration graphs*** - undirected graphs where two people are connected if they collaborate in a specific way
- ***influence graphs*** - directed graphs where there is an edge from one person to another if the first person can influence the second person

friendship graph: two people are connected if they are Facebook friends.



influence graph: who can influence whom?



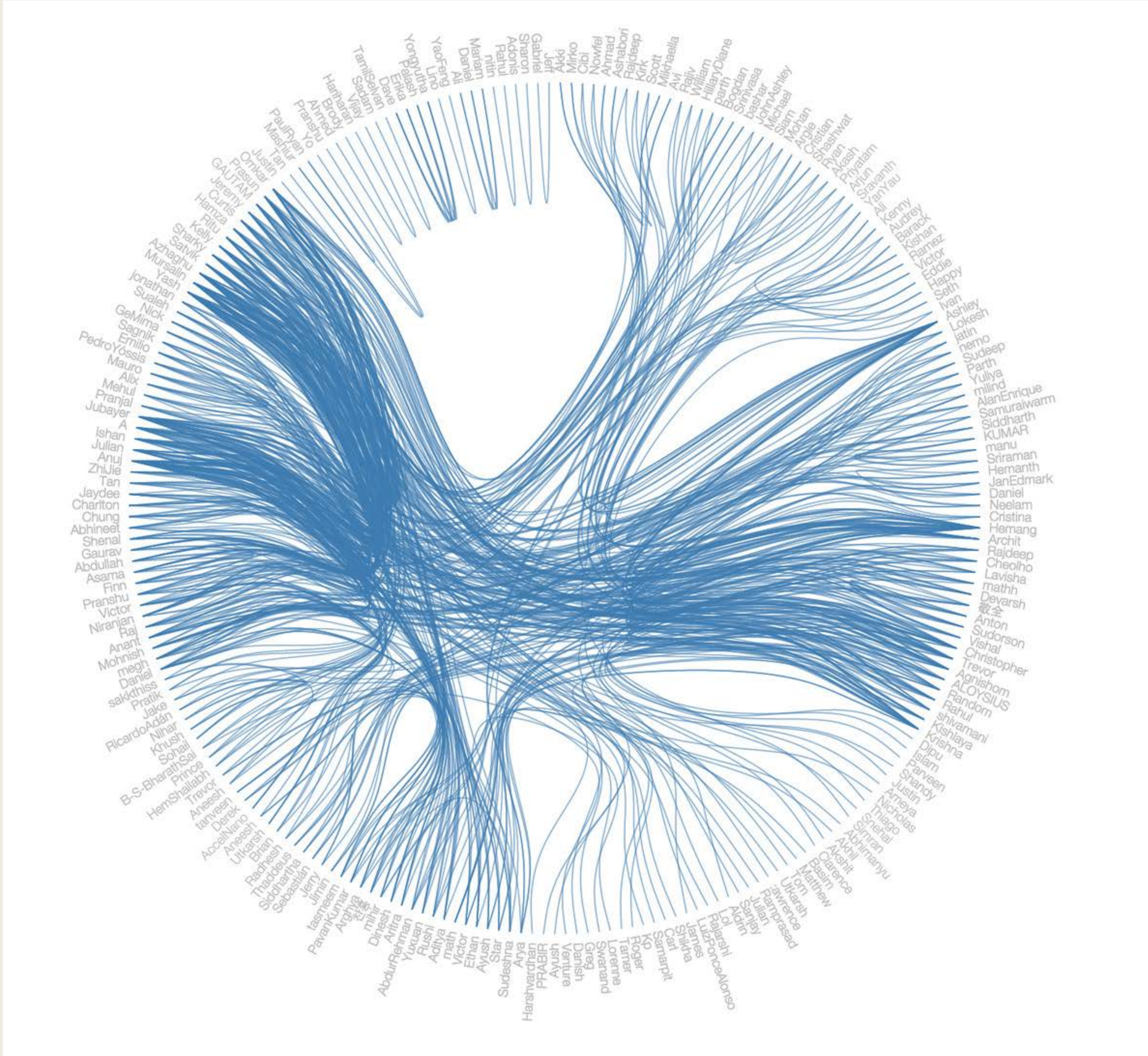
Collaboration Graphs

The ***Hollywood graph*** models the collaboration of actors.

- Represent actors by vertices and connect two vertices if the actors they represent have appeared in the same movie.
- Hollywood graphs are used when discussing Kevin Bacon numbers.

The ***academic collaboration*** *graph* models the collaboration of researchers who have coauthored a paper.

- Represent researchers as vertices. Connect the vertices representing two researchers if they are coauthors of a paper.
- Collaboration graph for mathematicians are used for *Erdős numbers*.



https://brilliant.org/wiki/wiki-collaboration-graph/

Graph Models: Information Networks

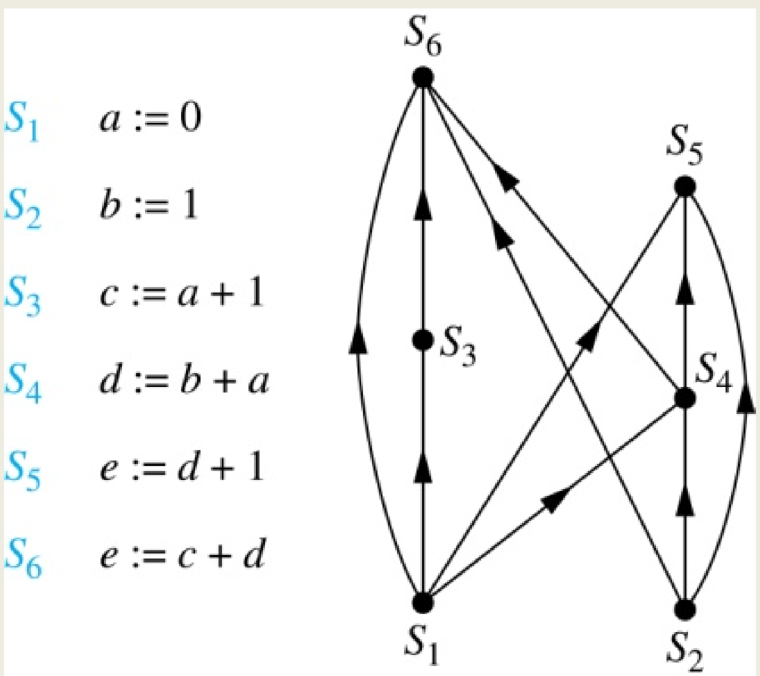
Graphs can be used to model networks that link different types of information.

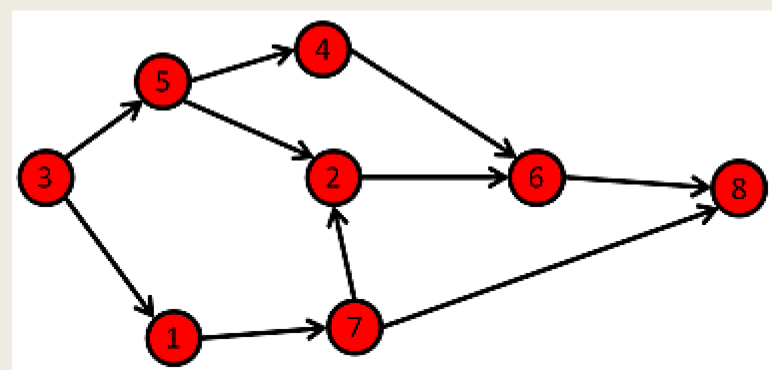
- In a ***web graph***, web pages are represented by vertices and links are represented by directed edges.
  - A web graph models the web at a particular time. Web graphs are used by search engines (11.4).
- In a ***citation network***:
  - Research papers in a particular discipline are represented by vertices.
  - When a paper cites a second paper as a reference, there is an edge from the vertex representing this paper to the vertex representing the second paper.

Graph Models: Execution of statements

Precedence graph: directed acyclic graph representing precedence relations; e.g., which statements must be executed before each statement, which course must be taken before another one.

- Vertices represent statements
- There is a directed edge from u to v when u needs to be executed before v.



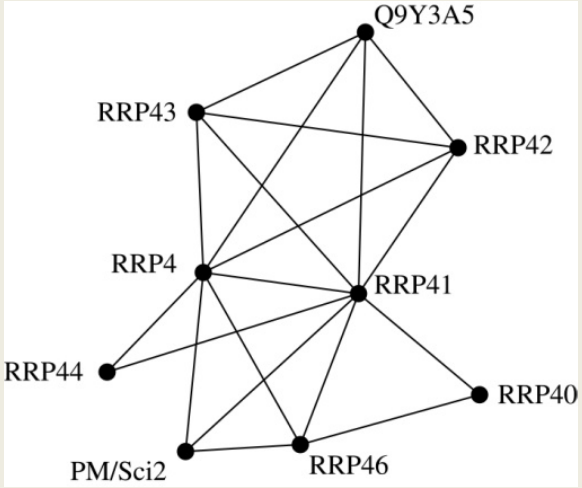


Biological Applications

Protein interaction graph: vertices represent proteins and vertices are connected by an edge if the proteins they represent interact.

- Protein interaction graphs are typically large; can contain more than 100,000 vertices and more than 1,000,000 edges.
- Protein interaction graphs are often split into smaller graphs, called *modules*, which represent the interactions between proteins involved in a particular function.

Module of the protein interaction graph of proteins that degrade RNA in a human cell.



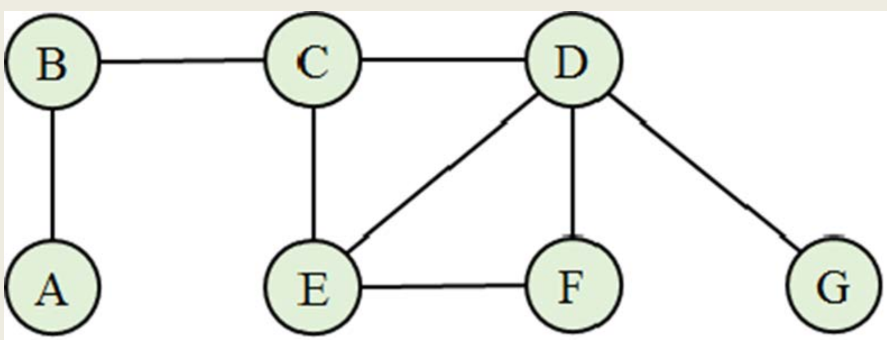
GRAPH TERMINOLOGY (KR 10.2)

- Graph terminology and basic graph properties
- Important types of graphs

Basic Terminology

Definition. Two vertices *u*, *v* in an undirected graph *G* are called ***adjacent*** (or *neighbors*) if there is an edge *e* between *u* and *v in G*.

Such an edge *e* is called *incident with* the vertices *u* and *v* and edge *e* is said to *connect u* and *v*.



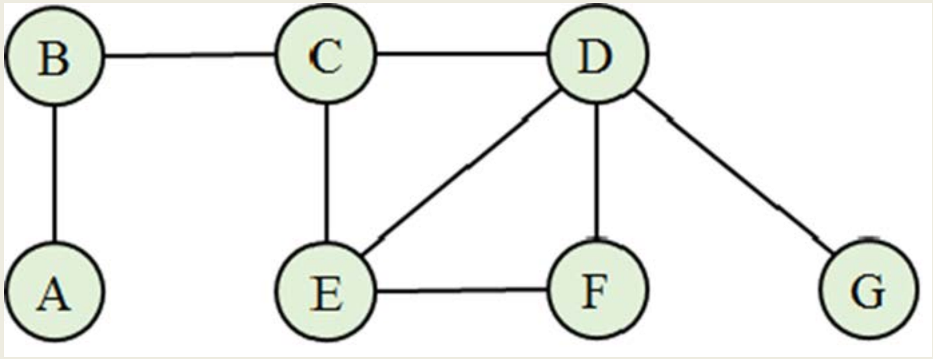
vertices C and D are adjacent
vertices C and F are not adjacent

Basic Terminology

Definition. The set of all neighbors of a vertex *v* of *G*=(*V*,*E*), denoted by *N*(*v*), is called the *neighborhood* of *v*.

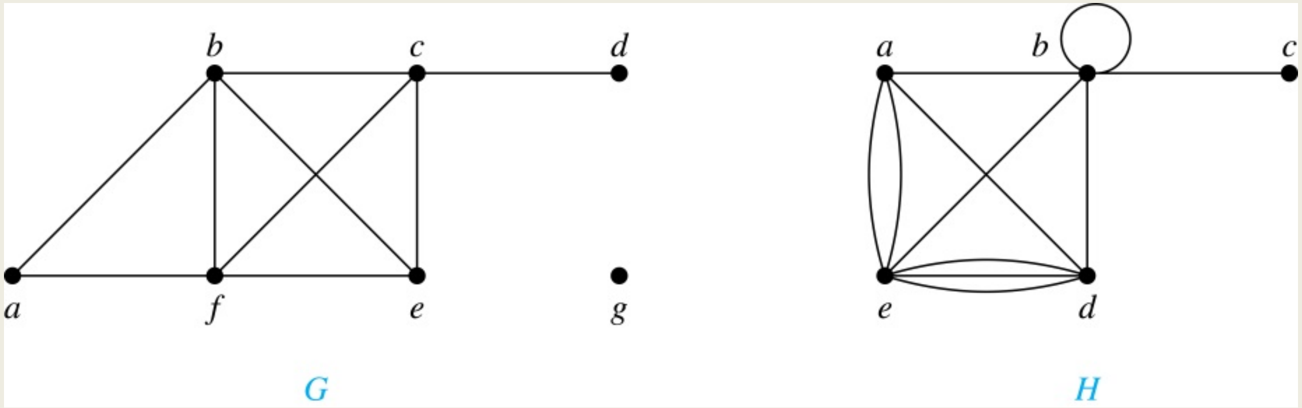
Definition. The ***degree*** *of a vertex in a undirected graph* is the number of edges incident with it.

A loop at a vertex contributes two to the degree of that vertex. The degree of the vertex *v* is denoted by deg(*v*).



deg(*D*) = 4

Degrees and Neighborhoods of Vertices



G: deg(a) = 2, deg(b) = deg(c) = deg(f) = 4, deg(d) = 1,

deg(e) = 3, deg(g) = 0.

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

$N(e) = \{b, c, f\}$, $N(f) = \{a, b, c, e\}$, $N(g) = \varnothing$

N({a,c,b}) = {a,b,c,d,e,f}

H: deg(a) = 4, deg(b) = deg(e) = 6, deg(c) = 1, deg(d) = 5.

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

$N(d) = \{a, b, e\}$, $N(e) = \{a, b, d\}$.

Theorem (*Handshaking Theorem*).

G = (*V*,*E*) is an undirected graph with *m* edges.

Then, $2m = \sum_{v \in V} \deg(v)$.

Proof:

Each edge contributes twice to the total degree count of all vertices. Hence, both the left-hand and right-hand sides of the equation equal twice the number of edges. ◀

Think about the graph where vertices represent the people at a party and an edge connects two people who have shaken hands.

Using the Handshaking Theorem

Example: How many edges are there in a graph consisting of 10 vertices, each of degree six?

Solution: Because the sum of the degrees of the vertices is $6 \cdot 10 = 60$, the handshaking theorem tells us that $2m = 60$. Hence, the number of edges is $m = 30$.

Example: If a graph has 5 vertices, can each vertex have degree 3?

Solution: This is not possible.
By the handshaking theorem, the sum of the degrees of the vertices is $3 \cdot 5 = 15$ which is odd.

Theorem: An undirected graph G =(V,E) has an even number of vertices of odd degree.

Proof: Let V_1 be the vertices of even degree and V_2 be the vertices of odd degree in graph *G.*

Assume G has *m* edges. Then,

$$2m = \sum_{v \in V} \deg(v) = \sum_{v \in V_1} \deg(v) + \sum_{v \in V_2} \deg(v).$$

even

Theorem: An undirected graph G =(V,E) has an even number of vertices of odd degree.

Proof: Let V_1 be the vertices of even degree and V_2 be the vertices of odd degree in graph *G.*
Assume G has *m* edges. Then,

$$2m = \sum_{v \in V} \deg(v) = \sum_{v \in V_1} \deg(v) + \sum_{v \in V_2} \deg(v).$$

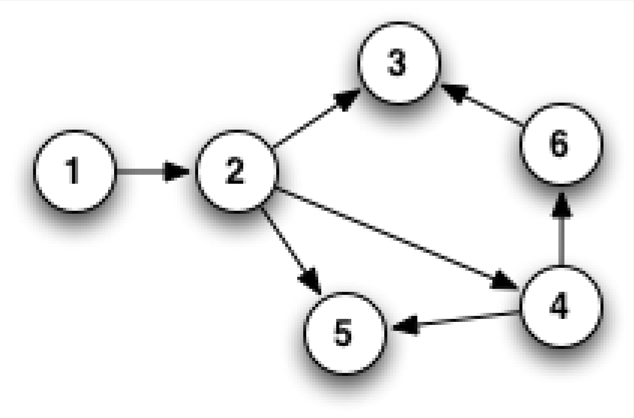
even

must be even since deg(*v*) is even for each $v \in V_1$

This sum must be even because (i) 2*m* is even and (ii) the sum of the degrees of the vertices of even degree is even.
Hence, the sum of the degrees of all vertices of odd degree is even and there must be an even number of such vertices.

Directed Graphs

Recall: In a *directed graph G* = (*V, E)* each edge is an ordered pair of vertices.

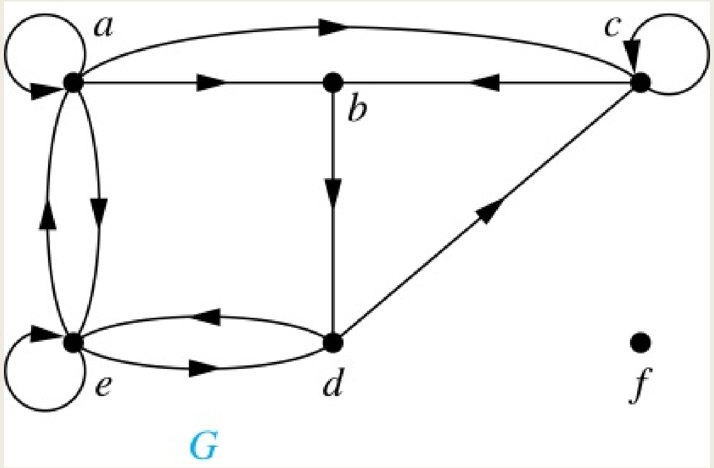


Definition:
Let (*u,v*) be an edge in *G*. Then, *u* is the *initial vertex* of this edge and u is *adjacent to v.*

The initial and terminal vertices of a loop are the same.

Definition:
The ***in-degree*** *of a vertex v*, denoted *deg*$^-$(*v*), is the number of edges which terminate at *v*.

The ***out-degree*** *of v*, denoted *deg*$^+$(*v*), is the number of edges with *v* as their initial vertex.



$\deg^-(a) = 2$, $\deg^-(b) = 2$, $\deg^-(c) = 3$,
$\deg^-(d) = 2$, $\deg^-(e) = 3$, $\deg^-(f) = 0$.

$\deg^+(a) = 4$, $\deg^+(b) = 1$, $\deg^+(c) = 2$,
$\deg^+(d) = 2$, $\deg^+(e) = 3$, $\deg^+(f) = 0$.

Directed Graphs

Theorem: Let *G* = (*V*, *E*) be a graph with directed edges. Then:

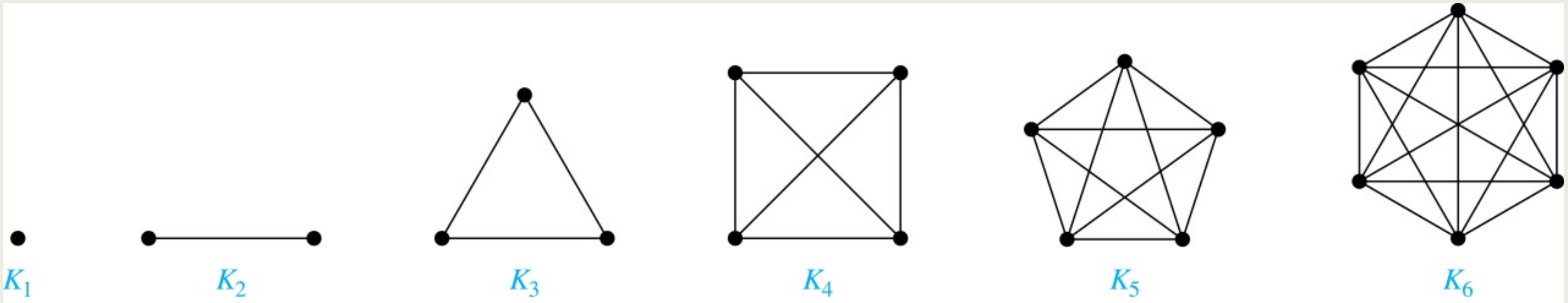
$$|E| = \sum_{v \in V} deg^-(v) = \sum_{v \in V} deg^+(v).$$

Proof: The first sum counts the number of outgoing edges over all vertices and the second sum counts the number of incoming edges over all vertices.
It follows that both sums equal the number of edges in the graph.

◀

Special Types of Simple Graphs: Complete Graphs

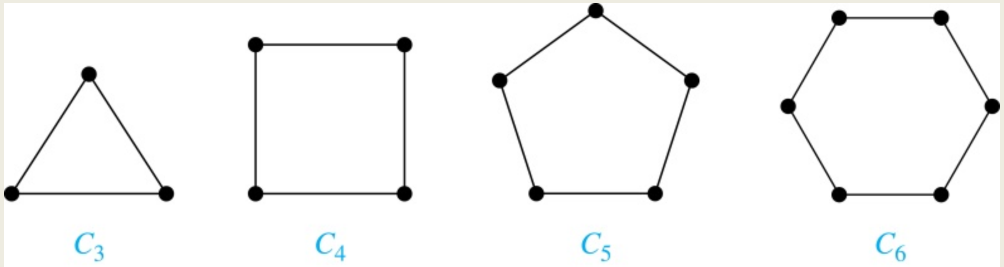
A *complete graph on n vertices*, denoted by K_n, is the simple graph that contains exactly one edge between each pair of distinct vertices.



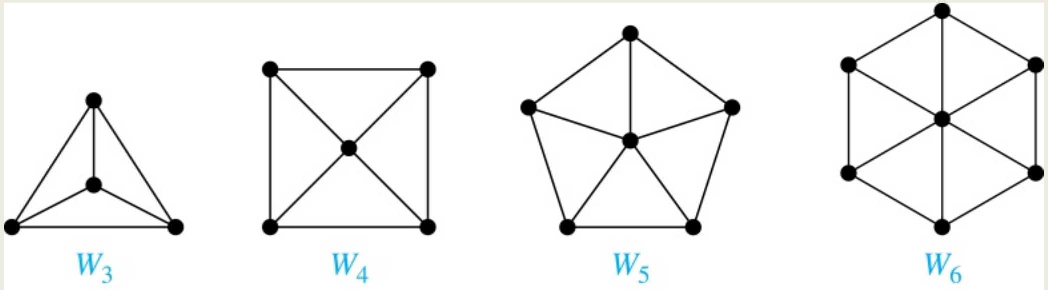
How many edges does a simple, complete graph with n vertices have?

Cycles and Wheels

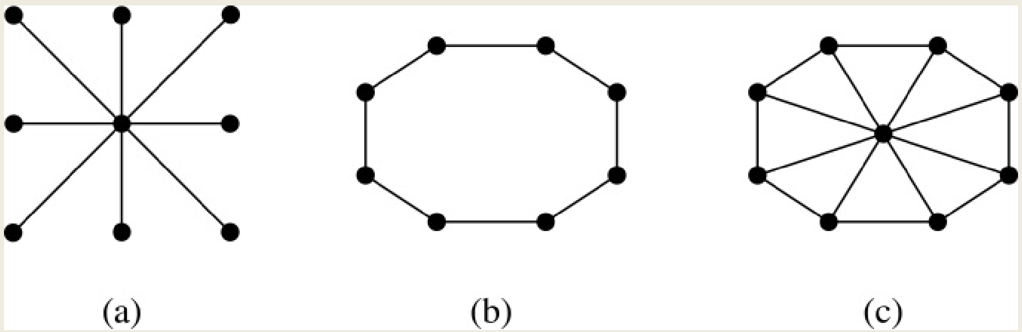
A ***cycle*** C_n for $n \geq 3$ consists of n vertices $v_1, v_2, \cdots, v_n$, and edges $\{v_1, v_2\}, \{v_2, v_3\}, \cdots, \{v_{n-1}, v_n\}, \{v_n, v_1\}$.

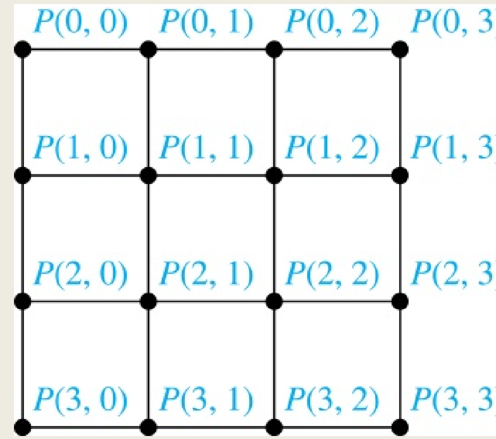


A ***wheel*** W_n is obtained by adding an additional vertex to a cycle C_n for $n \geq 3$ and connecting this new vertex to each of the n vertices in C_n by new edges.



Various special graphs play an important role in the design of computer networks.



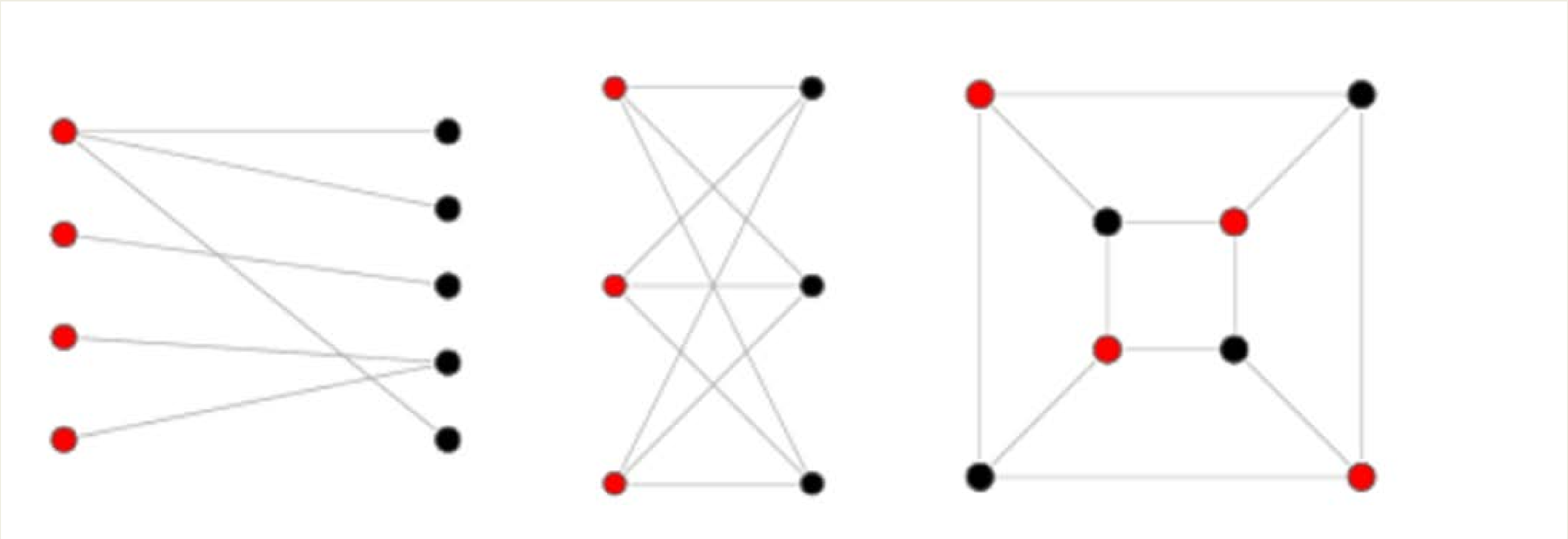


a) **Star topology**. All devices are connected to a central control device.
b) **Ring topology.**
c) **Wheel.** Combining the features of a star and a ring topology.
d) **Mesh network.**

Bipartite Graphs

Definition: A simple graph G is *bipartite* if V can be partitioned into two disjoint subsets V_1 and V_2 such that every edge connects a vertex in V_1 and a vertex in V_2.

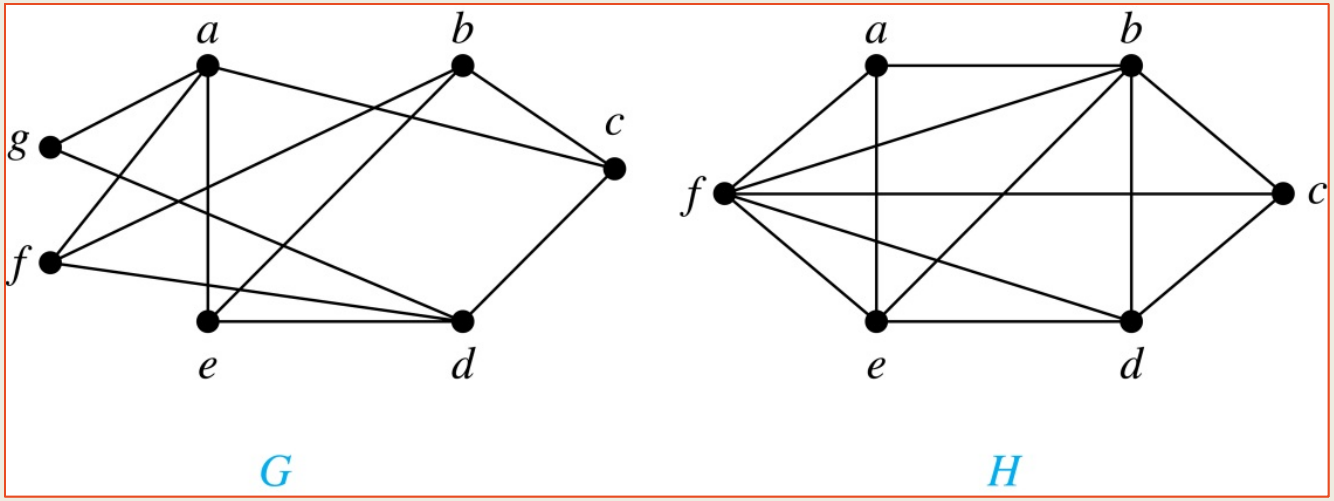
In other words, there are no edges which connect two vertices in V_1 or in V_2.



Bipartite Graphs

Equivalent definition of a bipartite graph:

A simple graph is bipartite iff it is possible to color the vertices red or blue so that no two adjacent vertices have the same color.



G is bipartite

H is not bipartite

Theorem. A simple graph $G=(E,V)$ is bipartite iff it is possible to color the vertices red or blue so that no two adjacent vertices have the same color.

Proof: (→) Assume G is bipartite. Then, let V_1 and V_2 be the sets as specified in the definition. If we assign color red to the vertices in V_1 and blue to the vertices in V_2, then no two adjacent vertices have the same color.

(←) Assume the vertices can be colored red and blue as stated above. Let V_1 be the set of red vertices and V_2 be the set of blue vertices. V_1 and V_2 are disjoint and their union is V. Furthermore, every edge in G connects a vertex in V_1 with a vertex in V_2. It follows that G is bipartite.

REPRESENTING GRAPHS (KR 10.3)

- Adjacency lists and matrices

Representing Graphs: Adjacency Lists

Definition: An *adjacency list* specifies the vertices that are adjacent to each vertex of the graph.

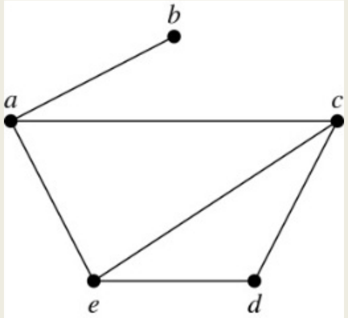


TABLE 1 An Adjacency List for a Simple Graph.

Vertex	*Adjacent Vertices*
a	b, c, e
b	a
c	a, d, e
d	c, e
e	a, c, d

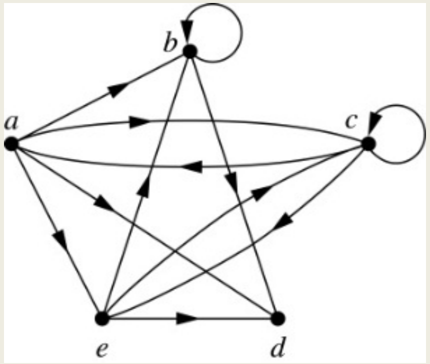


TABLE 2 An Adjacency List for a Directed Graph.

Initial Vertex	*Terminal Vertices*
a	b, c, d, e
b	b, d
c	a, c, e
d	
e	b, c, d

Representation of Graphs: Adjacency Matrices

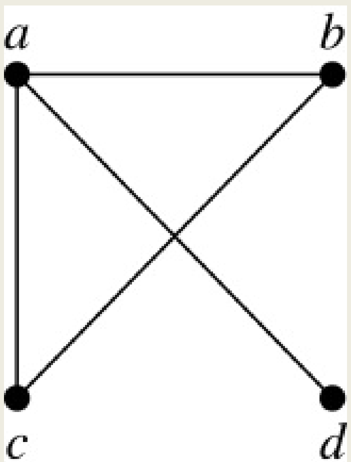
Definition: Suppose that $G = (V, E)$ is a simple graph where $|V| = n$. Arbitrarily list the vertices of G as $v_1, v_2, \ldots, v_n$.

The *adjacency matrix* $\mathbf{A}_G$ of G, with respect to the listing of vertices, is the $n \times n$ zero-one matrix with 1 as its (i, j)-th entry when v_i and v_j are adjacent, and 0 as its (i, j)-th entry when they are not adjacent.

In other words, if the graphs adjacency matrix is $\mathbf{A}_G = [a_{ij}]$, then

$$a_{ij} = \begin{cases} 1 & \text{if } \{v_i, v_j\} \text{ is an edge of } G, \\ 0 & \text{otherwise.} \end{cases}$$

The vertex ordering is a, b, c, d.



$$\begin{array}{c c} & \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{array}{c} a \\ b \\ c \\ d \end{array} & \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Directed: Count only outgoing vertices.

When a graph is sparse, that is, it has few edges relatively to the total number of possible edges, it is typically more efficient to represent the graph using an adjacency list than an adjacency matrix.

Note: The adjacency matrix of a simple graph is symmetric, i.e., $a_{ij} = a_{ji}$. Since there are no loops, we have $a_{ii} = 0$.

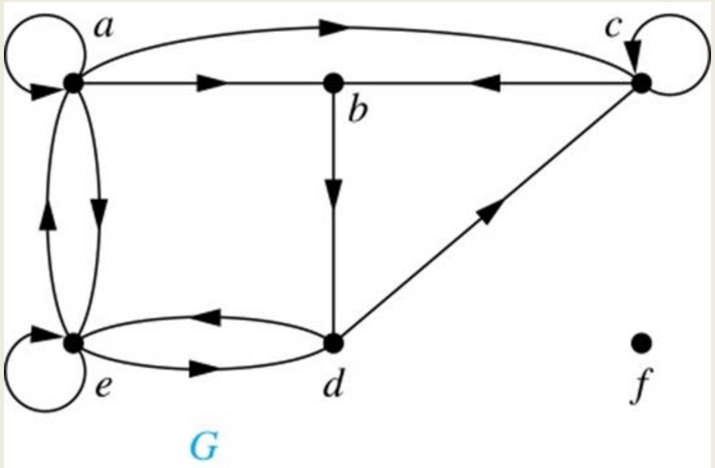
Adjacency Matrices: Directed graphs

The matrix for a directed graph $G=(V, E)$ has a 1 in its (i, j)-th position if there is an edge from v_i to v_j, where V = $\{v_1, v_2, \dots v_n\}$.

In other words, if the graphs adjacency matrix is $\mathbf{A}_G = [a_{ij}]$, then

$$a_{ij} = \begin{cases} 1 & \text{if } \{v_i, v_j\} \text{ is an edge of } G, \\ 0 & \text{otherwise.} \end{cases}$$

The adjacency matrix for a directed graph does not have to be symmetric.



Graphs, Data Structures and Algorithms

Graphs in algorithms

- Names of vertices are mapped to integers 1, 2, …, n
- Edges often have weights
- Vertices often maintain additional entries
- Most algorithms assume simple graphs
- Performance of an algorithm is typically impacted by the graph representation.

Adjacency matrix: 2-dimensional array

Adjacency lists: 1-dimensional array with each entry associated with a linked list

CONNECTIVITY (KR 10.4)

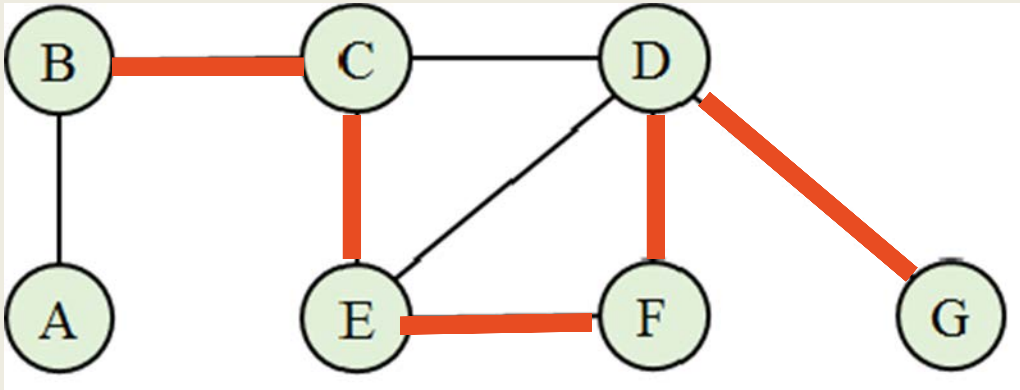
- Paths
- Shortest Paths
- Counting paths
- Connected components

Paths in undirected graphs

A *path* is a sequence of edges that begins at a vertex and travels from vertex to vertex along edges of the graph.

Numerous problems can be modeled with paths formed by traveling along edges of graphs such as:

- determining whether a message can be sent between two computers.
- efficiently planning routes for mail delivery.



Definition: A *path* of *length k* from u to v in G is a sequence of k edges $e_1, \ldots, e_k$ for which there exists a sequence $x_0 = u, x_1, \ldots, x_{k-1}, x_k = v$ of vertices such that edge e_i has the endpoints x_{i-1} and x_i, for $i = 1, \ldots, k$.

- When the graph is simple, we denote this path by its vertex sequence $x_0, x_1, \ldots, x_k$.
- The path is a *circuit (cycle)* if it begins and ends at the same vertex (i.e., $u = v$).
- A path or circuit is *simple* if it does not contain the same edge more than once.

simple path / circuit: Does not contain the same edge more than once.

Definition is readily extended to directed graphs.

Example: Paths

In the simple graph here:

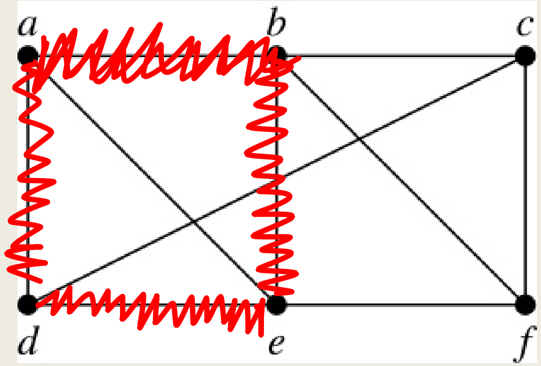
- *a*, *d*, *c*, *f*, *e* is a simple path of length 4.
- *d*, *e*, *c*, *a* is not a path because *e* is not connected to *c*.
- *b*, *c*, *f*, *e*, *b* is a circuit of length 4.
- *a*, *b*, *e*, *d*, *a*, *b* is a path of length 5, but it is not a simple path.

→ # edges in a path

Path Denoted as:

$V_1, V_2, \ldots\ldots, V_n$

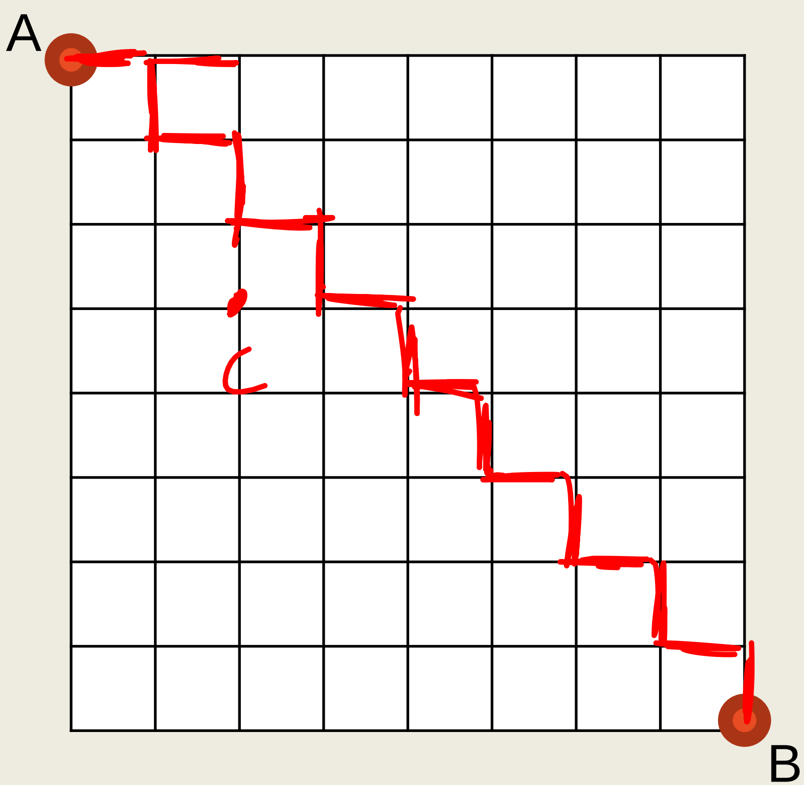
where V_n is the n^{th} vertex



Counting Paths

Graph G is a 9 by 9 grid graph (81 vertices).

How many different shortest paths exist from A to B?



$$\frac{16!}{8!\,8!}$$

$$\frac{16!}{8!\,8!}$$

$$\frac{5!}{3!\,2!}$$

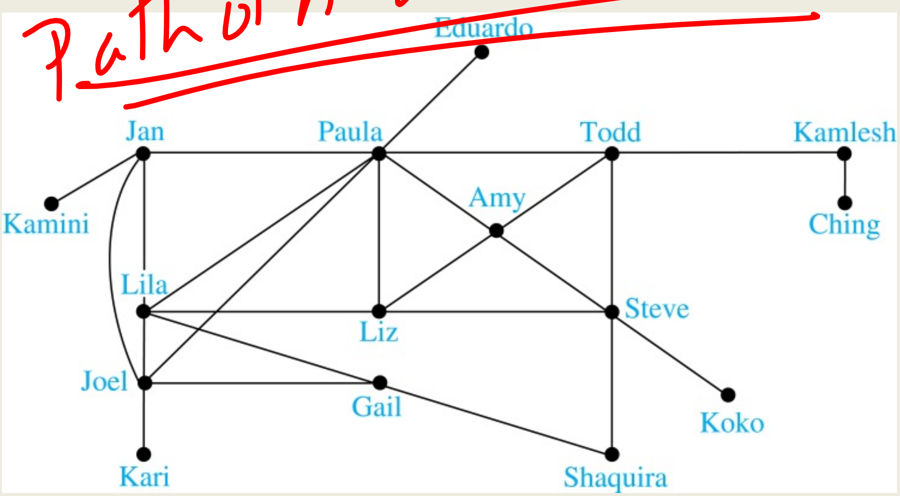
$$\binom{5}{3}$$

Degrees of Separation

Paths in Acquaintanceship Graphs: There is a path between two people if there is a chain of people linking these people, where two people adjacent in the chain know one another.

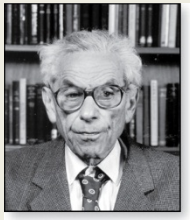
In the graph below, there is a chain of six people linking Kamini and Ching.

Path of Acquainteship Graph



Some have speculated that almost every pair of people in the world are linked by a small chain of no more than six, or maybe even, five people

Erdős numbers



Paul Erdős
(1913-96)

In a collaboration graph, two people *A* and *B* are connected by a path when there is a sequence of people starting with *A* and ending with *B* such that the endpoints of each edge in the path are people who have collaborated.

In the academic collaboration graph of people who have written papers in mathematics, the ***Erdős number*** of a person *M* is the length of the *shortest path* between *M* and the prolific mathematician Paul Erdős.

TABLE 1 The Number of Mathematicians with a Given Erdős Number (as of early 2006).

Erdős Number	*Number of People*
0	1
1	504
2	6,593
3	33,605
4	83,642
5	87,760
6	40,014
7	11,591
8	3,146
9	819
10	244
11	68
12	23
13	5

Bacon Numbers

Kevin Bacon Number	# of People
0	1
1	3452
2	403921
3	1504560
4	390201
5	34150
6	4181
7	601
8	119
9	7
10	1

In the Hollywood graph, two actors *A* and *B* are linked when there is a chain of actors linking *A* and *B*, where every two actors adjacent in the chain have acted in the same movie.

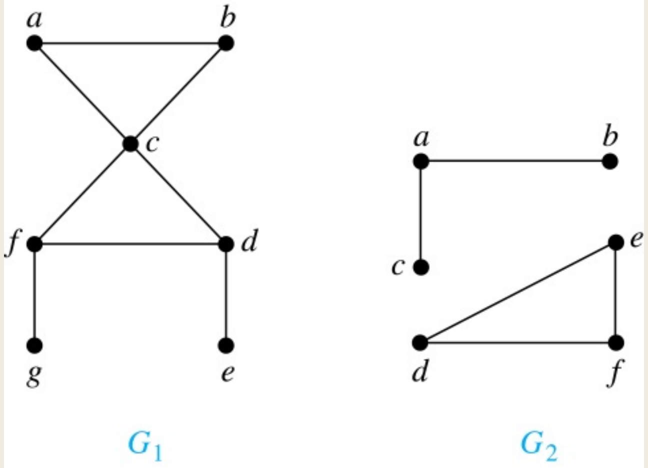
- The *Bacon number* of an actor *C* is defined to be the length of the *shortest path* connecting *C* and the well-known actor Kevin Bacon.
- The *oracle of Bacon* web site http://oracleofbacon.org/how.php provides a tool for finding Bacon numbers.

Undirected Graph $\longrightarrow$ Connected.

Connectedness in Undirected Graphs

Definition: An undirected graph is called ***connected*** if there is a path between every pair of vertices.
An undirected graph that is not *connected* is called *disconnected*.

Example: G_1 is connected because there is a path between any pair of its vertices. G_2 is not connected (there is no path between vertices *a* and *f*, for example).



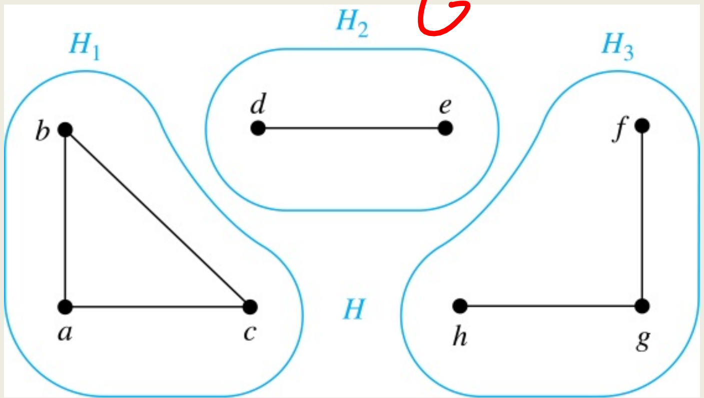
Connected Components

Definition: A ***connected component*** of a graph G is a connected subgraph of G that is not a proper subgraph of another connected subgraph of G.

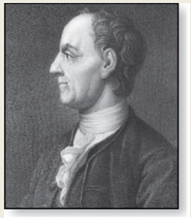
A graph G that is not connected has two or more connected components that are disjoint and have G as their union.

Example: The graph H is the union of three disjoint subgraphs H_1, H_2, and H_3. These three subgraphs are the connected components of H.

6



Euler Paths and Circuits



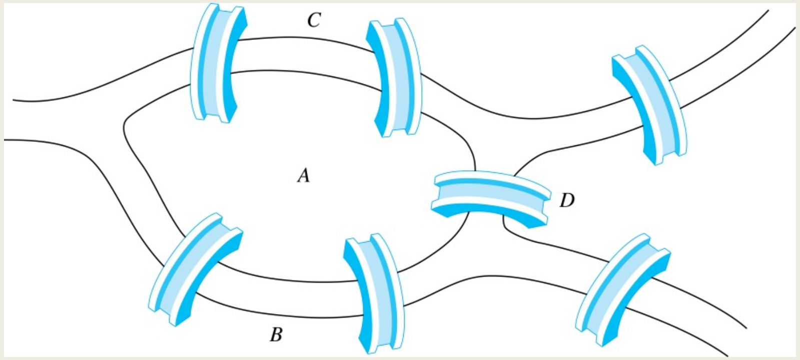
Leonard Euler
(1707-1783)

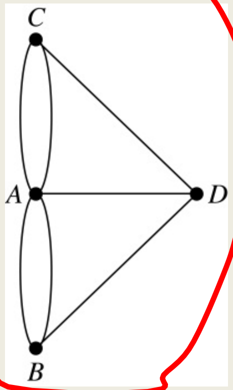
The town of Kӧnigsberg, Prussia (now Kalingrad, Russia) was divided into four sections by the branches of the Pregel river. In the 18th century seven bridges connected these regions.

People wondered whether it was possible to follow a path that crosses each bridge exactly once and returns to the starting point.

The Swiss mathematician Leonard Euler proved that no such path exists. This result is often considered to be the first theorem ever proved in graph theory.

The 7 Bridges of Kӧnigsberg





Multigraph Model

Euler circuits

A given path is considered to be an euler cycle if each vertex has an even degree.

Definition: An Euler circuit in an undirected graph G is a simple circuit containing every edge of G.

When does a graph have an Euler circuit?
What are the necessary and sufficient conditions?

If G has an Euler circuit, every vertex must have an even degree.

Why?

Assume every vertex in G has an even degree. Does an Euler circuit exist?

Yes. Proof by construction.

Examples of graph problems/algorithms

Connected components
Depth-first search traversal
Breadth first search traversal
Stronger forms of connectivity (bi-, bridge, etc)
Path finding
Shortest paths
Longest path
Hamiltonian path
Minimum cost spanning tree
Graph coloring
Graph isomorphism

Graphs can be huge, infinite, dynamically changing; can have uncertainties

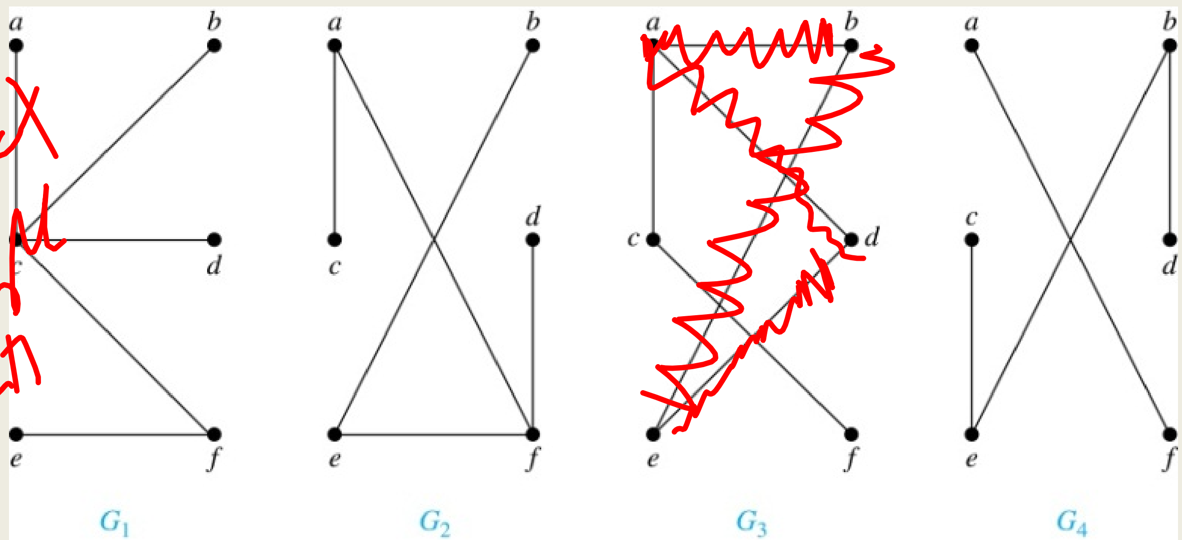
TREES (KR 11.1 + EXAMPLES)

- Rooted trees, ordered trees
- Properties of trees
- Recursion on trees

Trees

A *tree* is a connected, undirected graph with no simple circuits

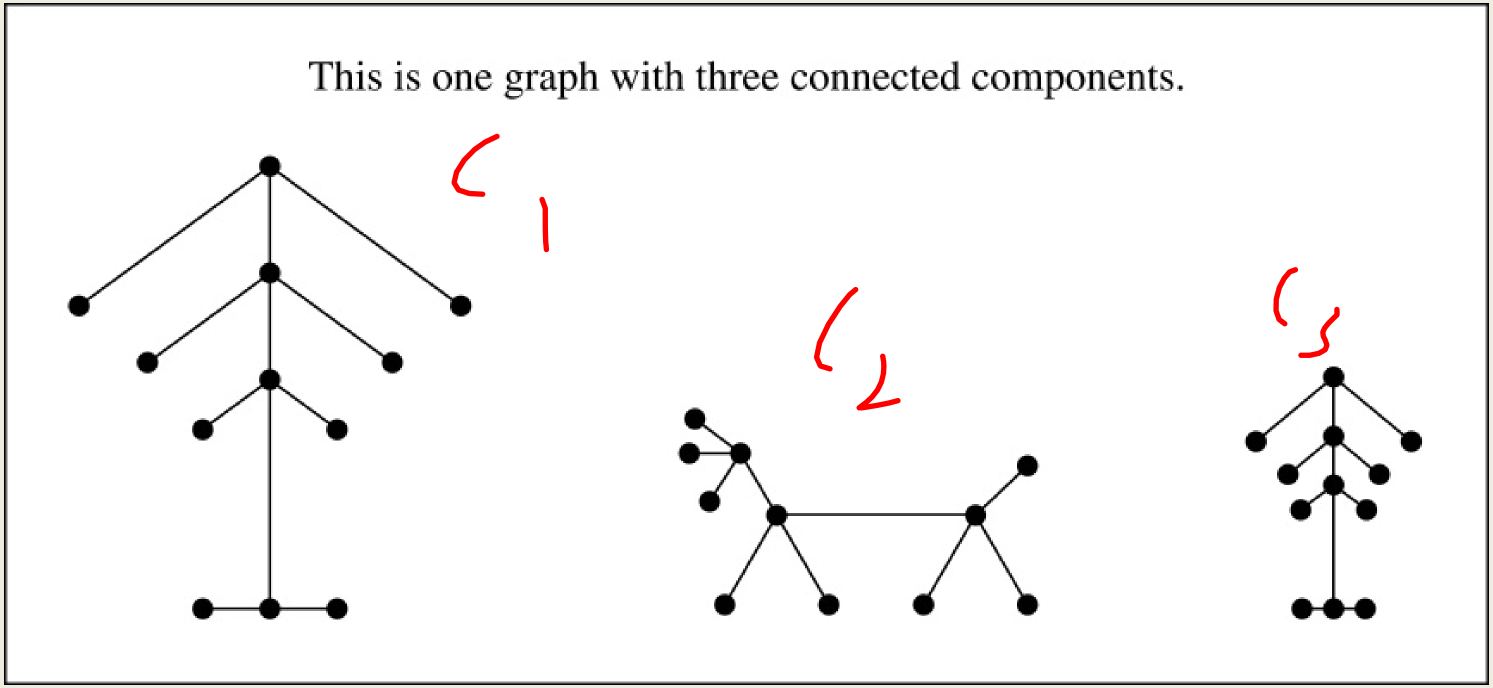
connected
no simple
circuit



- G_1 and G_2 are trees; both are connected and have no simple circuits.
- G_3 is not a tree as *e*, *b*, *a*, *d*, *e* is a simple circuit
- G_4 is not a tree because it is not connected

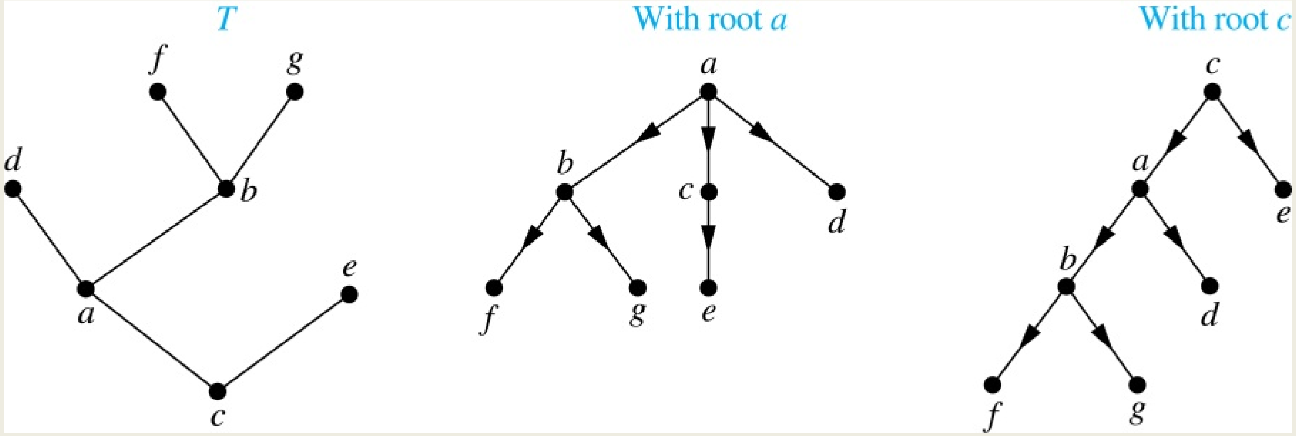
Forest

A *forest* is a graph that has no simple circuit, but is not connected. Each of the connected components in a forest is a tree



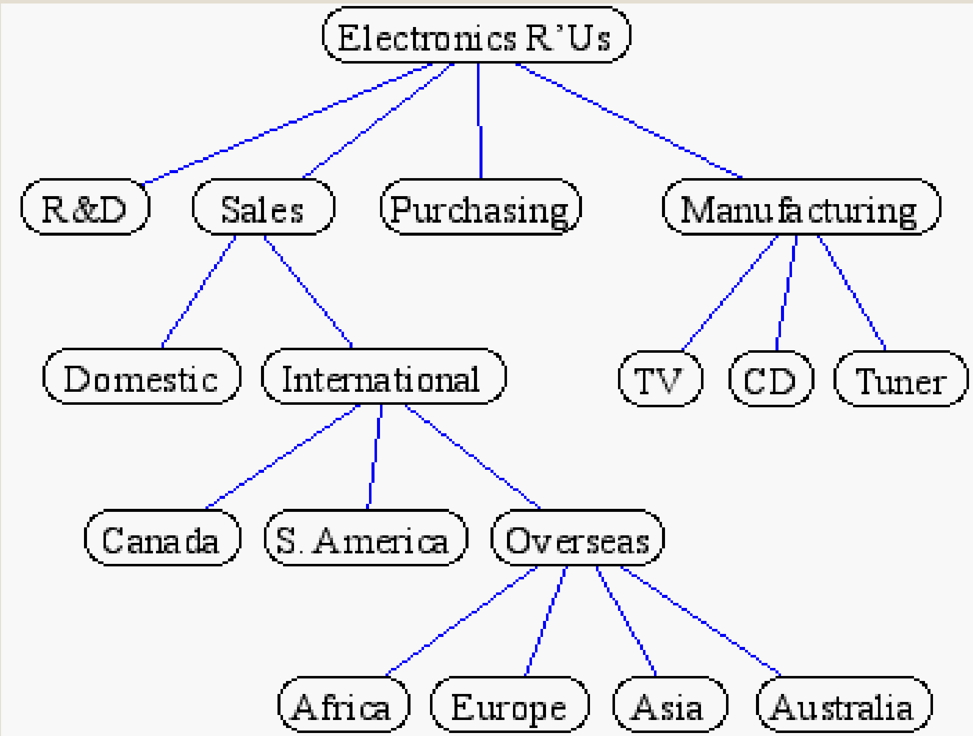
Rooted Trees

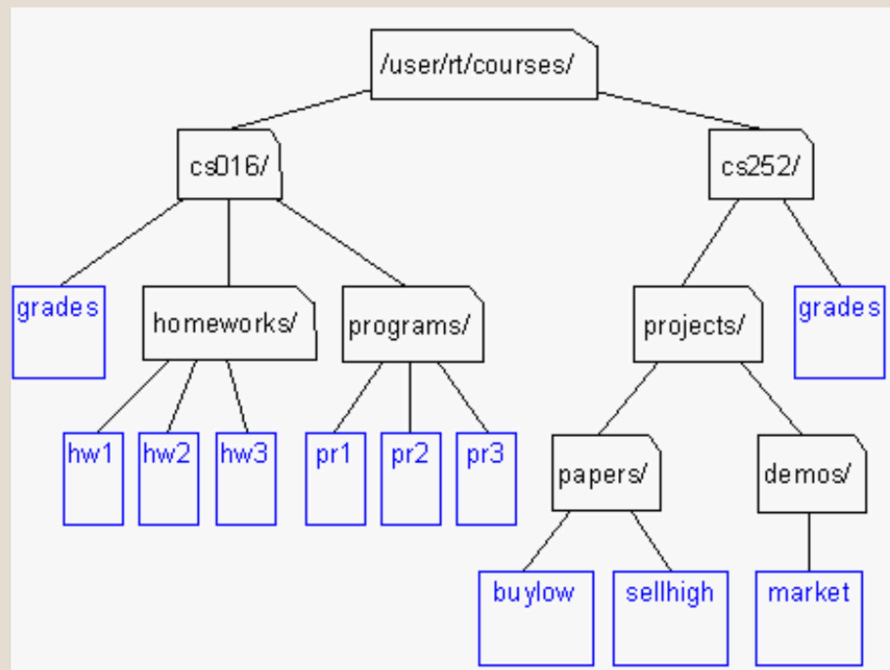
A *rooted tree* is a tree in which one vertex has been designated as the *root* and every edge is directed away from the root

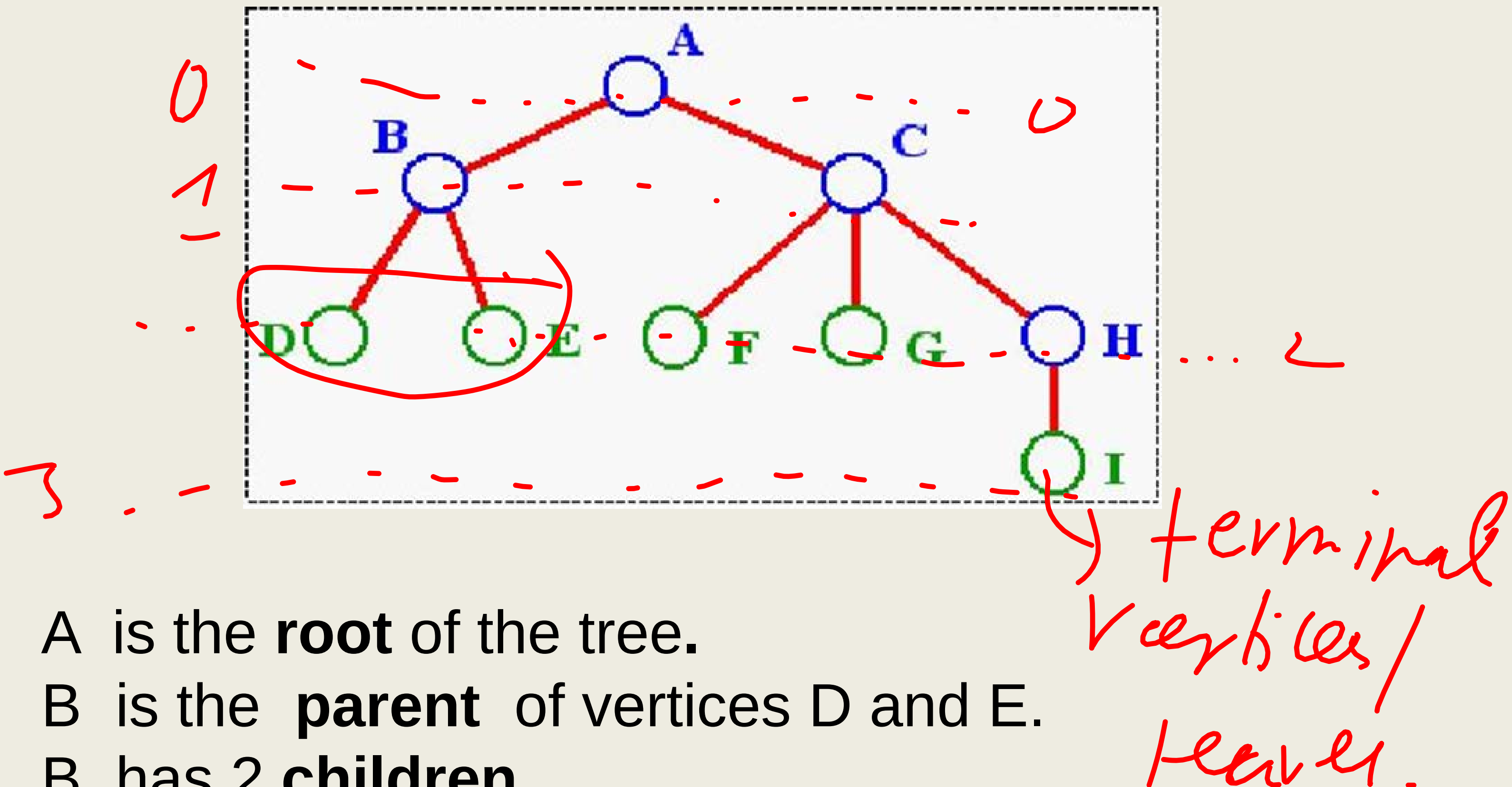


An unrooted (free) tree is converted into different rooted trees when a vertex is chosen as the root

Rooted trees can represent a hierarchy (e.g., file system, organization structure of a corporation).







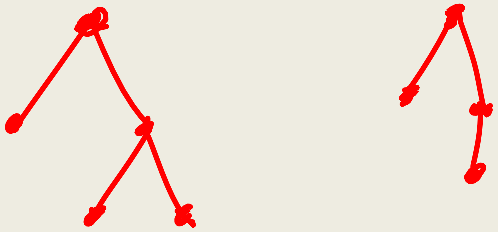
A is the **root** of the tree.
B is the **parent** of vertices D and E.
B has 2 **children**.
C is the **sibling** of vertex B.
D, E, F, G, and I are **terminal vertices** or **leaves**.
Vertex E is on **level** 2 (root A is at level 0).
The **height** of the tree is 3.

m-ary Rooted Trees

A rooted tree is called an *m-ary tree* if every internal vertex has no more than *m* children.

The tree is called a *full m-ary tree* if every internal vertex has exactly *m* children.

An *m*-ary tree with *m* = 2 is called a *binary* tree.



Theorem: A tree with *n*, n≥1, vertices has *n* − 1 edges.

2 proofs …

Thm: A tree with n vertices has
n − 1 edges.

Theorem: A tree with n vertices, n≥1, has $n - 1$ edges.

Proof (*by mathematical induction*):

BASIS STEP: *A* tree with one vertex has no edges. Hence, the theorem holds when $n = 1$.

INDUCTIVE STEP: Assume that every tree with k vertices has $k-1$ edges.

Suppose tree T has $k + 1$ vertices and that v is a leaf of T. Let vertex w be the parent of v.

Removing the vertex v and the edge connecting w to v produces a tree T' with k vertices.
By the inductive hypothesis, T' has $k - 1$ edges.
Because T has one more edge than T', we see that T has k edges.

The claim follows.

Theorem: A tree with *n* vertices, $n \geq 1$, has $n - 1$ edges.

Proof:

Create a rooted version of the tree with an arbitrary vertex r as the root.

Create a one-to-one correspondence of vertices to edges (except for root r):

- associate vertex u with parent v with edge (v,u)

This makes n-1 associations and covers all edges of the tree.

Hence, a tree with *n* vertices has $n - 1$ edges. ◀

Trees as data structures

- Binary search trees
- Decision trees
- Prefix codes
- Tries

Algorithms on trees

- Pre, post, inorder traversal + more
- Breadth-first search