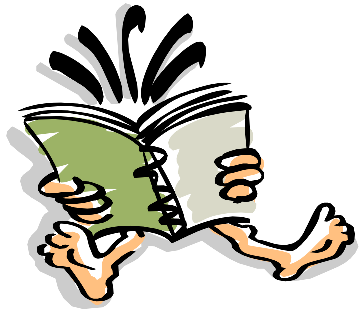
Discrete Structures 2

Chapter 10: Graphs



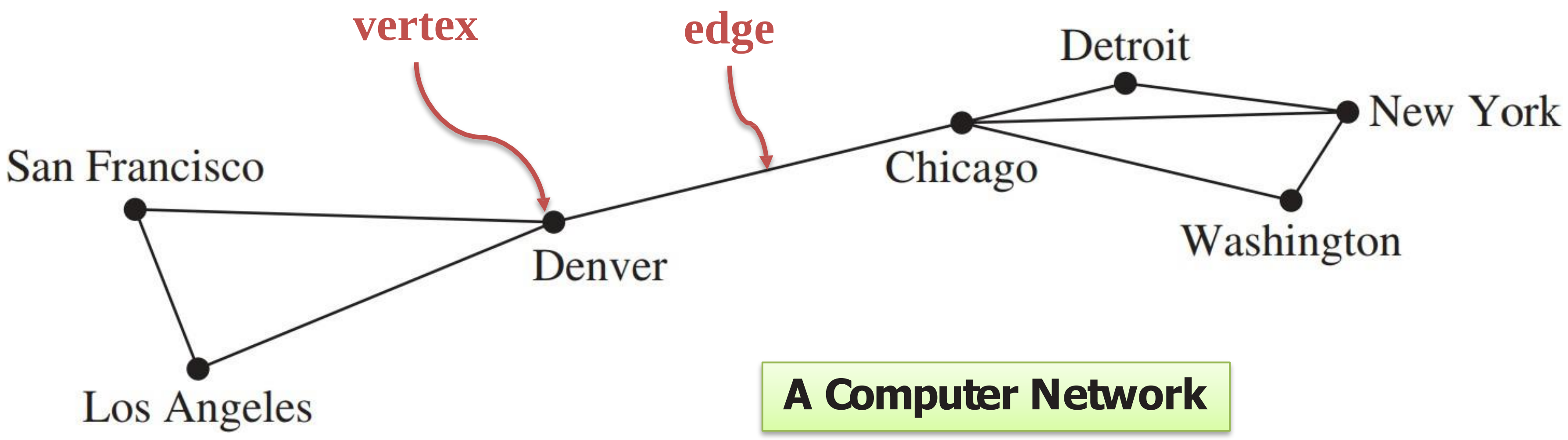
Chapter 10: Graphs

- Graphs and Graph Models.
- Graph Terminology and Special Types of Graphs.
- Representing Graphs and Graph Isomorphism.
- Connectivity.
- Shortest-Path Problems.

Graphs and Graph Models (1/29)

Definition:

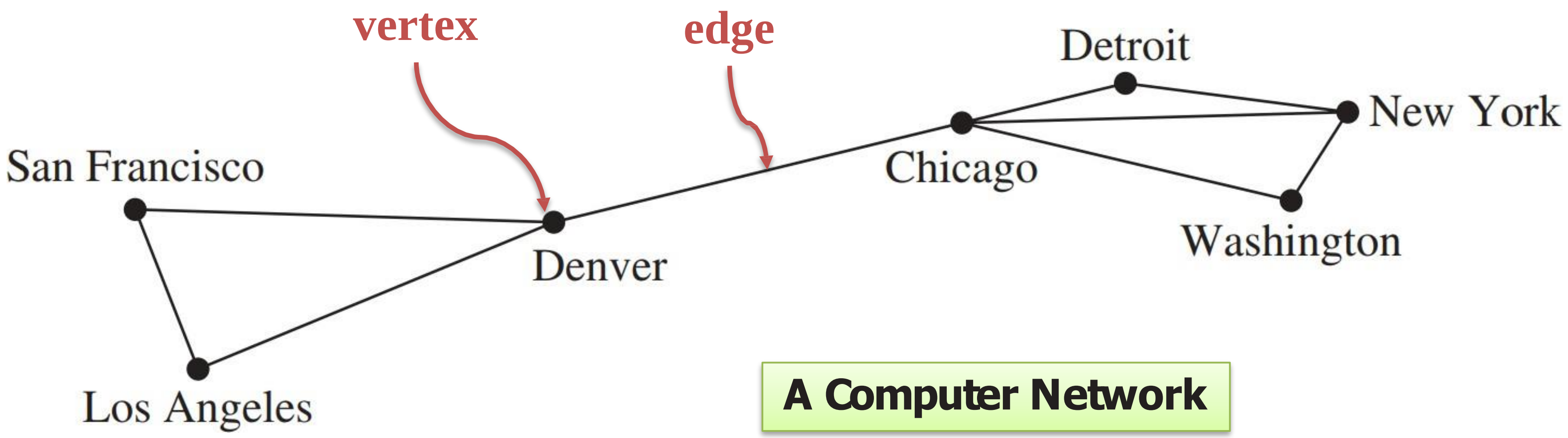
A graph $G = (V, E)$ consists of V, a nonempty set of **vertices** (or **nodes**) and E, a set of **edges**. Each edge has either one or two vertices associated with it, called its **endpoints**. An edge is said to **connect** its endpoints.



Graphs and Graph Models (2/29)

Remark:

The set of vertices V of a graph G may be infinite. A graph with an *infinite vertex set* or an *infinite number of edges* is called an **infinite graph**, and in comparison, a graph with a *finite vertex* set and a *finite edge* set is called a **finite graph**. In this chapter, we will usually consider only finite graphs.

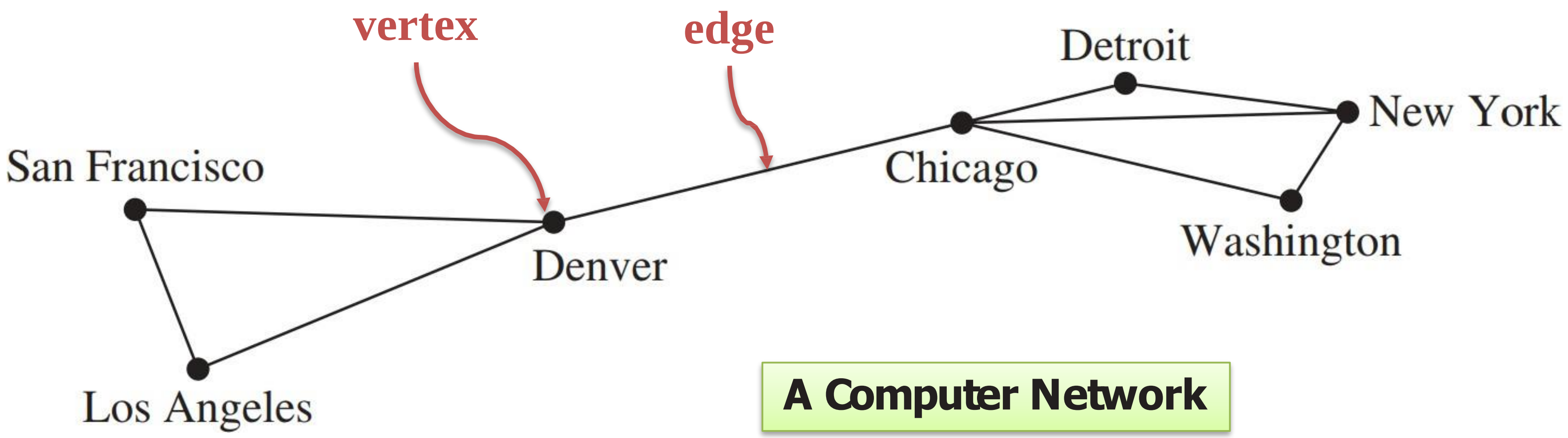


A Computer Network

Graphs and Graph Models (3/29)

Simple Graph:

Note that each edge of the graph representing this computer network connects two different vertices. A graph in which *each edge connects two different vertices* and where no two edges connect the same pair of vertices is called a ***simple graph***.

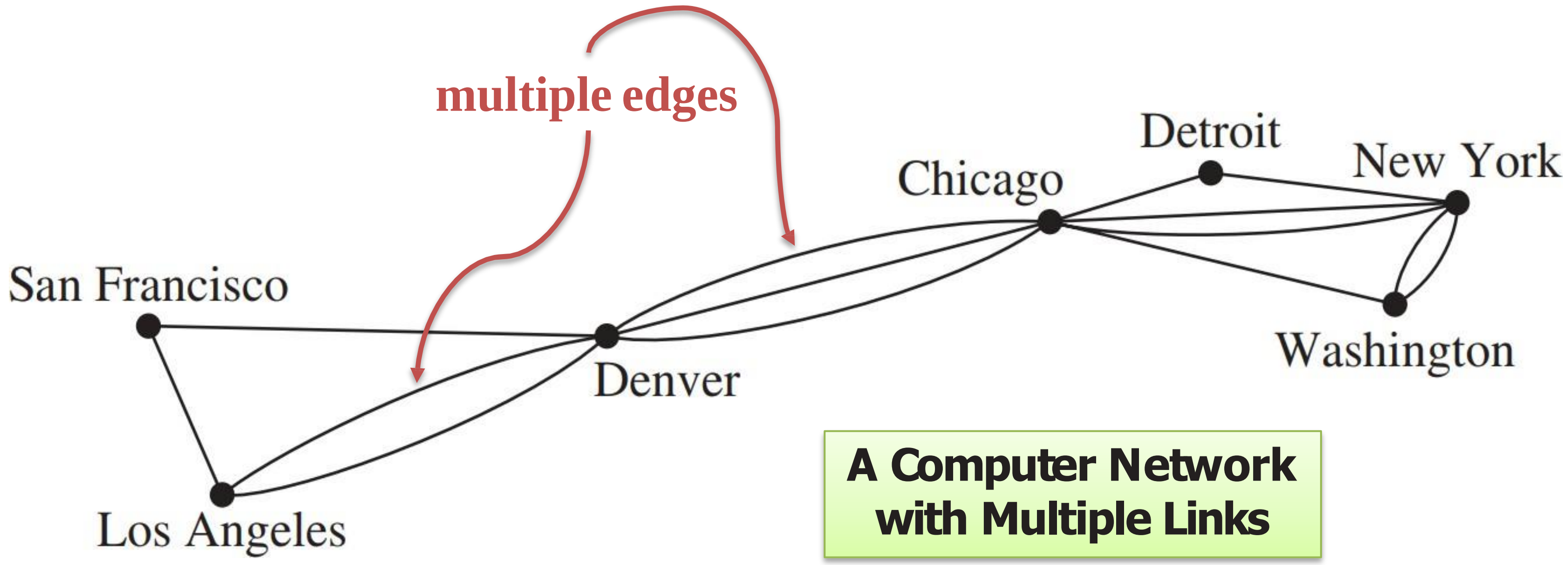


A Computer Network

Graphs and Graph Models (4/29)

Multigraphs:

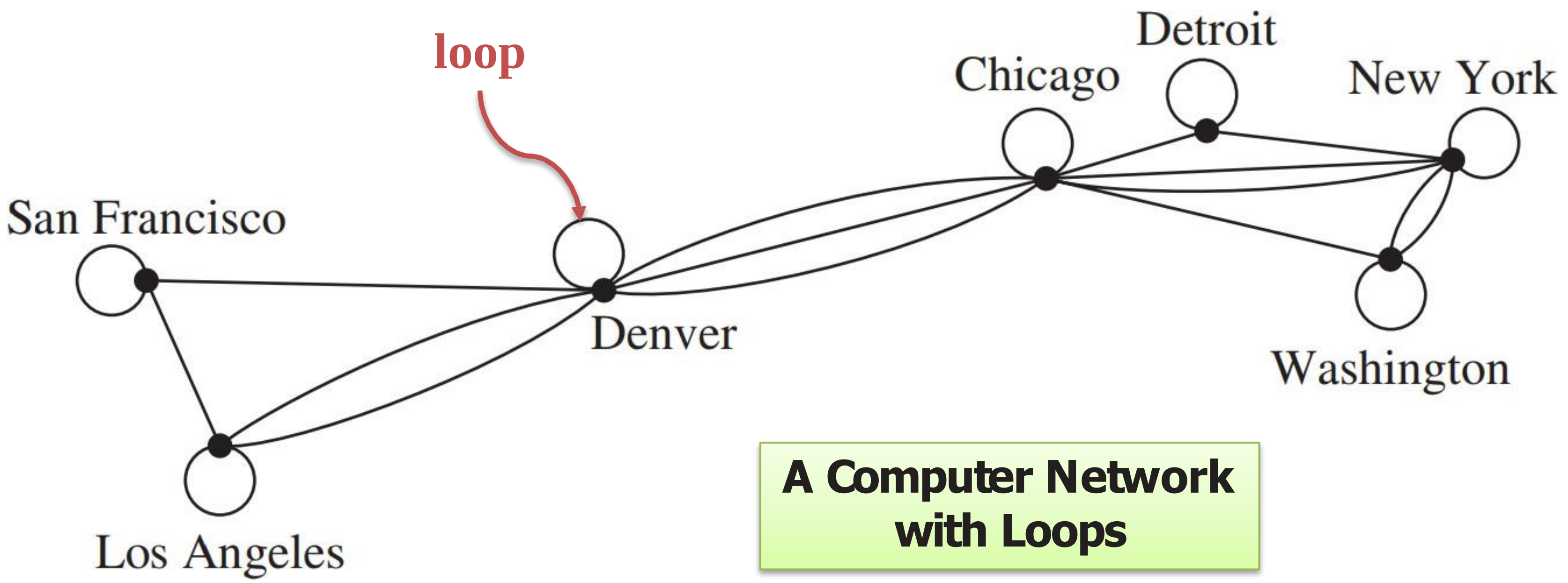
Graphs that may have multiple edges connecting the same vertices are called ***multigraphs***.



Graphs and Graph Models (5/29)

Loop:

Edges that connect a vertex to itself are called ***loops***.

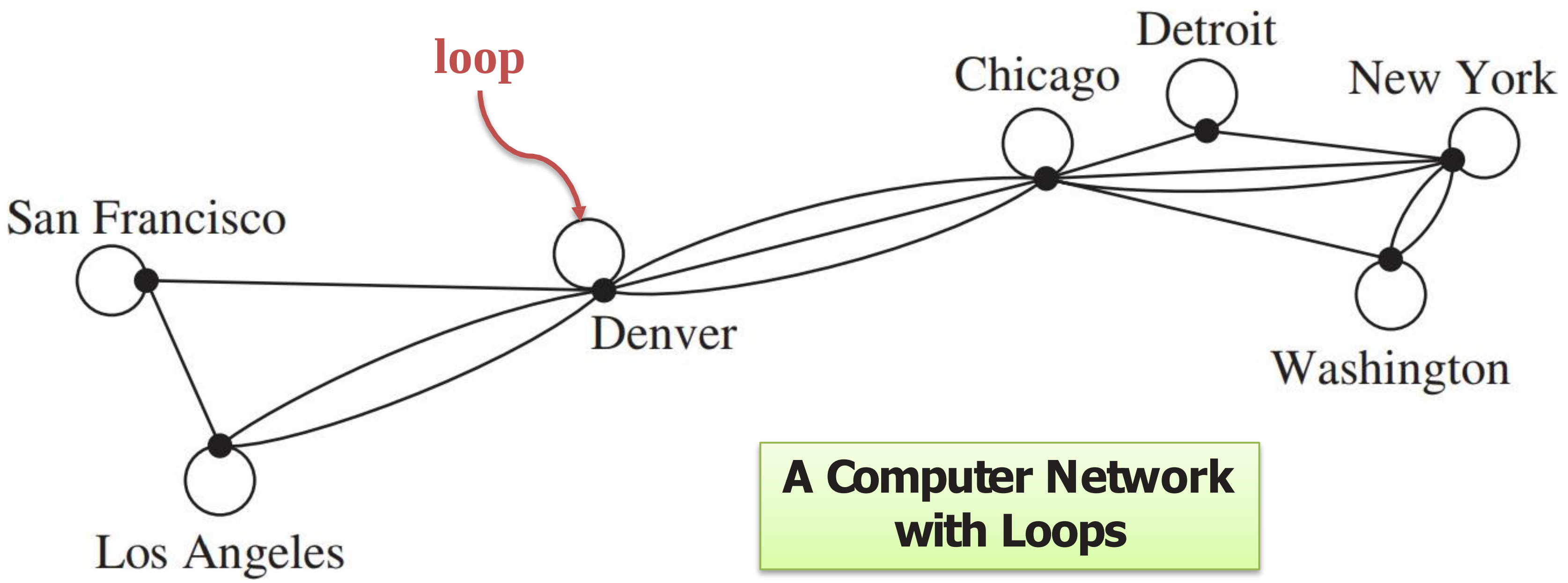


A Computer Network with Loops

Graphs and Graph Models (6/29)

Pseudographs:

Graphs that may include loops, and possibly multiple edges connecting the same pair of vertices or a vertex to itself, are sometimes called ***pseudographs***.

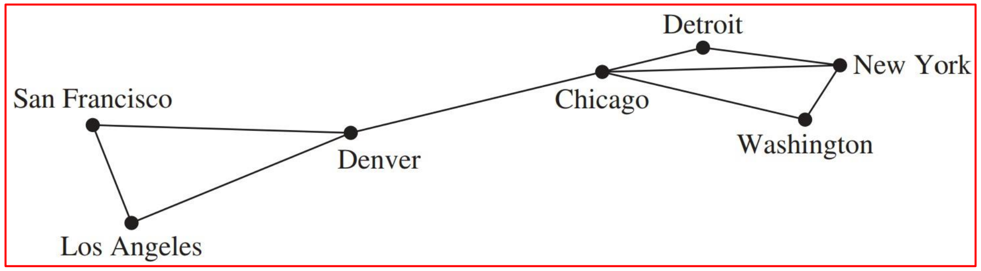


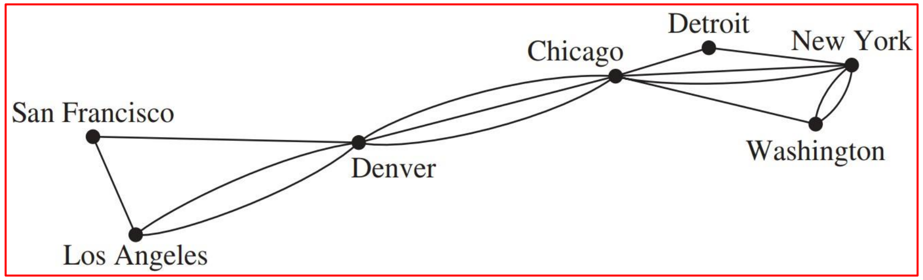
A Computer Network with Loops

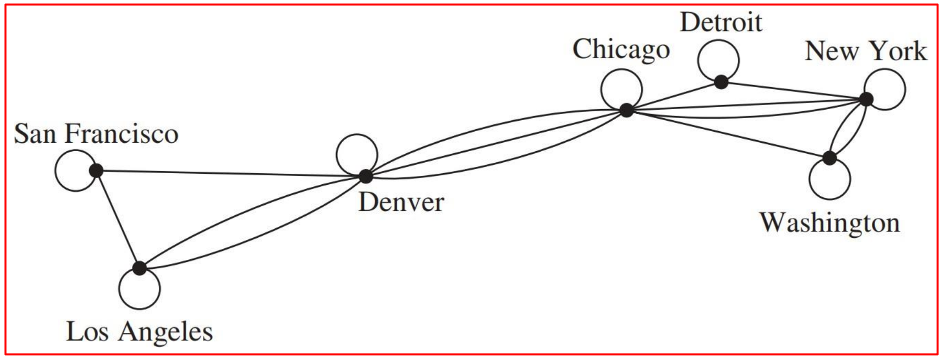
Graphs and Graph Models (7/29)

Undirected Graphs:

Undirected edges



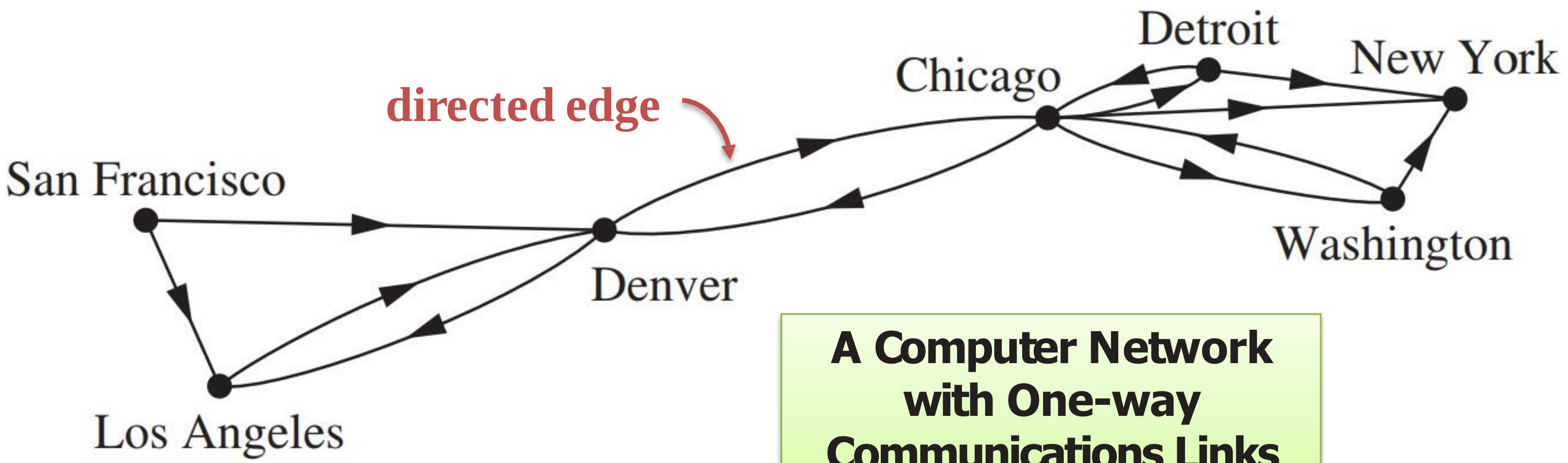




Graphs and Graph Models (8/29)

Definition:

A ***directed graph*** (or ***digraph***) (V, E) consists of a nonempty set of vertices V and a set of *directed edges* (or *arcs*) E. Each directed 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.

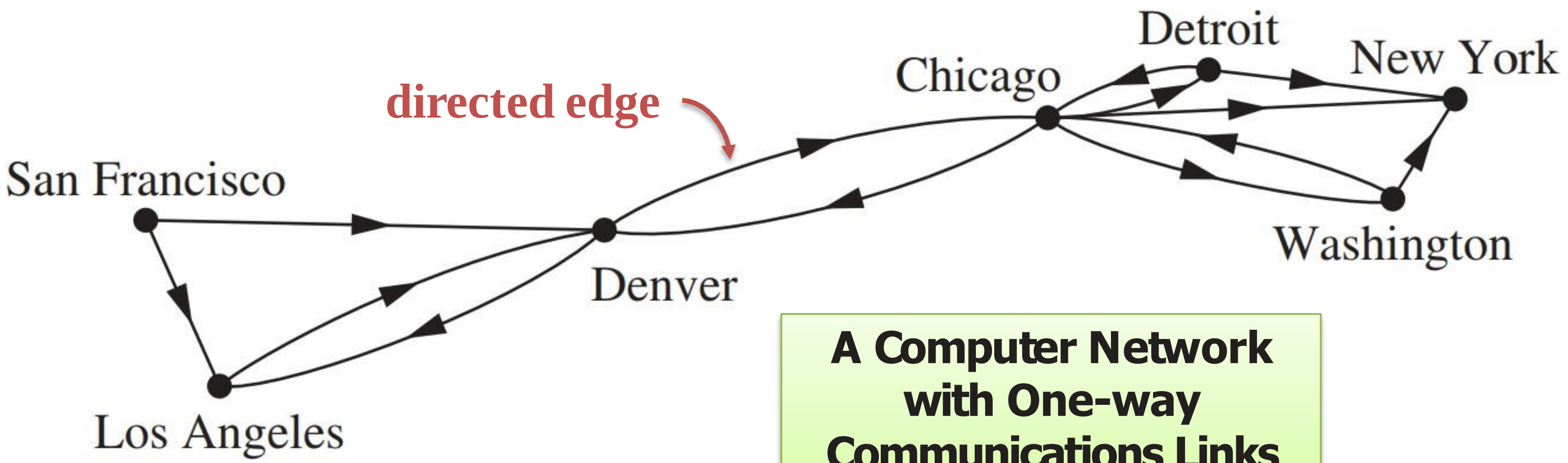


A Computer Network with One-way Communications Links

Graphs and Graph Models (9/29)

Simple Directed Graph:

When a directed graph has ***no*** *loops* and has ***no*** *multiple directed edges*, it is called a ***simple directed graph***.

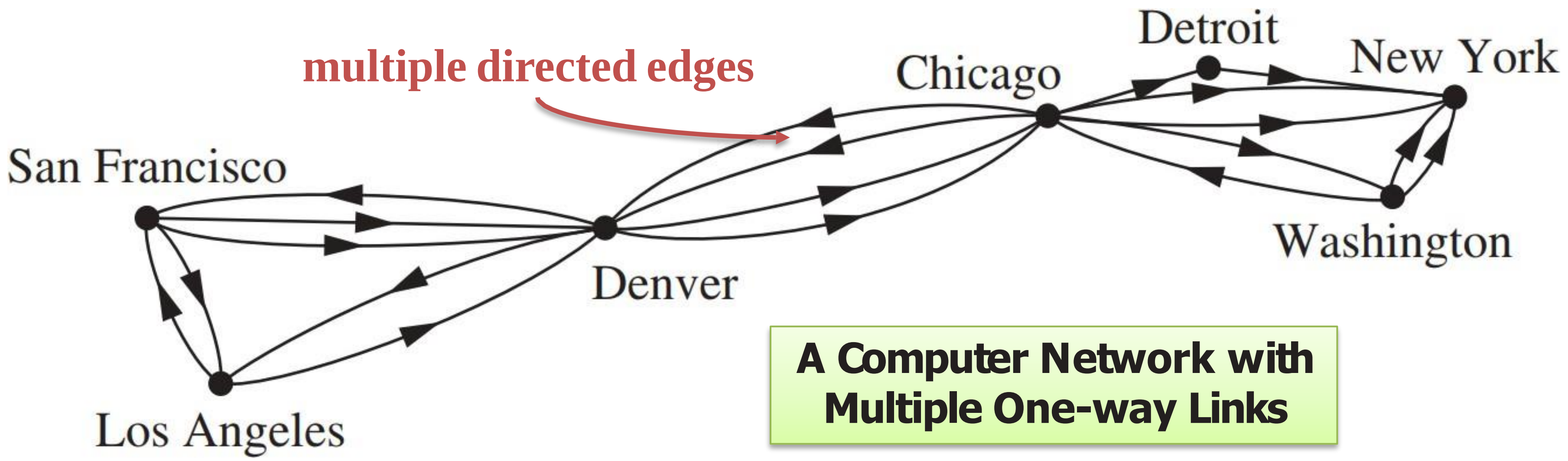


A Computer Network with One-way Communications Links

Graphs and Graph Models (10/29)

Directed Multigraphs:

Directed graphs that may have *multiple directed edges* from a *vertex* to a *second* (possibly the same) *vertex* are used to model such networks. We called such graphs ***directed multigraphs***.

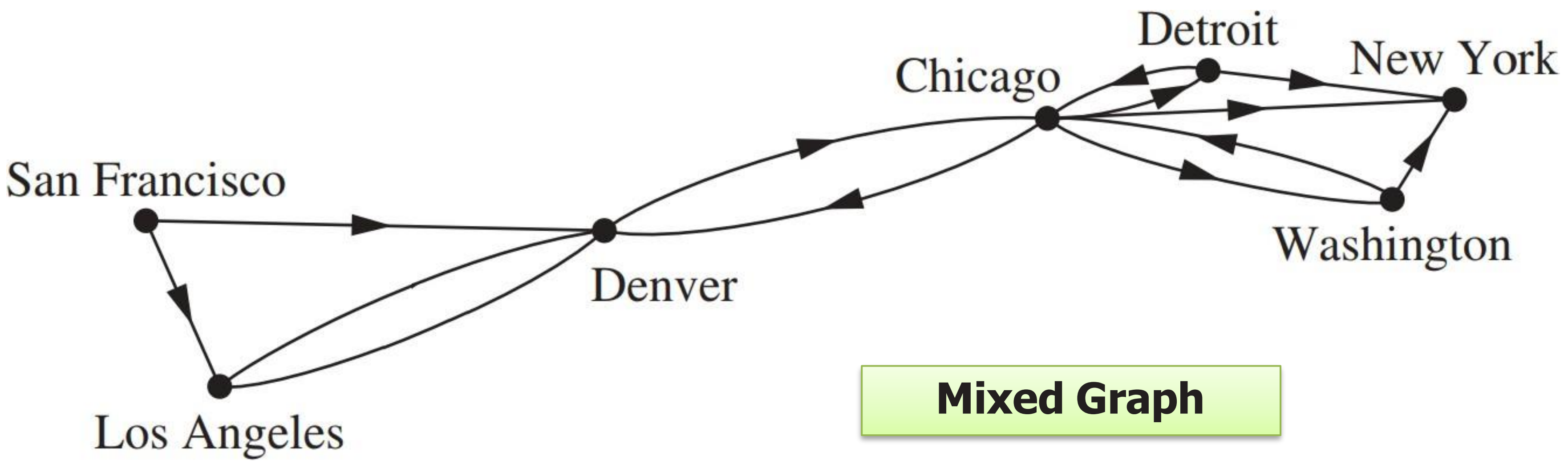


A Computer Network with Multiple One-way Links

Graphs and Graph Models (11/29)

Mixed Graph:

For some models we may need a graph where some edges are undirected, while others are directed. A graph with both directed and undirected edges is called a ***mixed graph***.



Mixed Graph

Graphs and Graph Models (12/29)

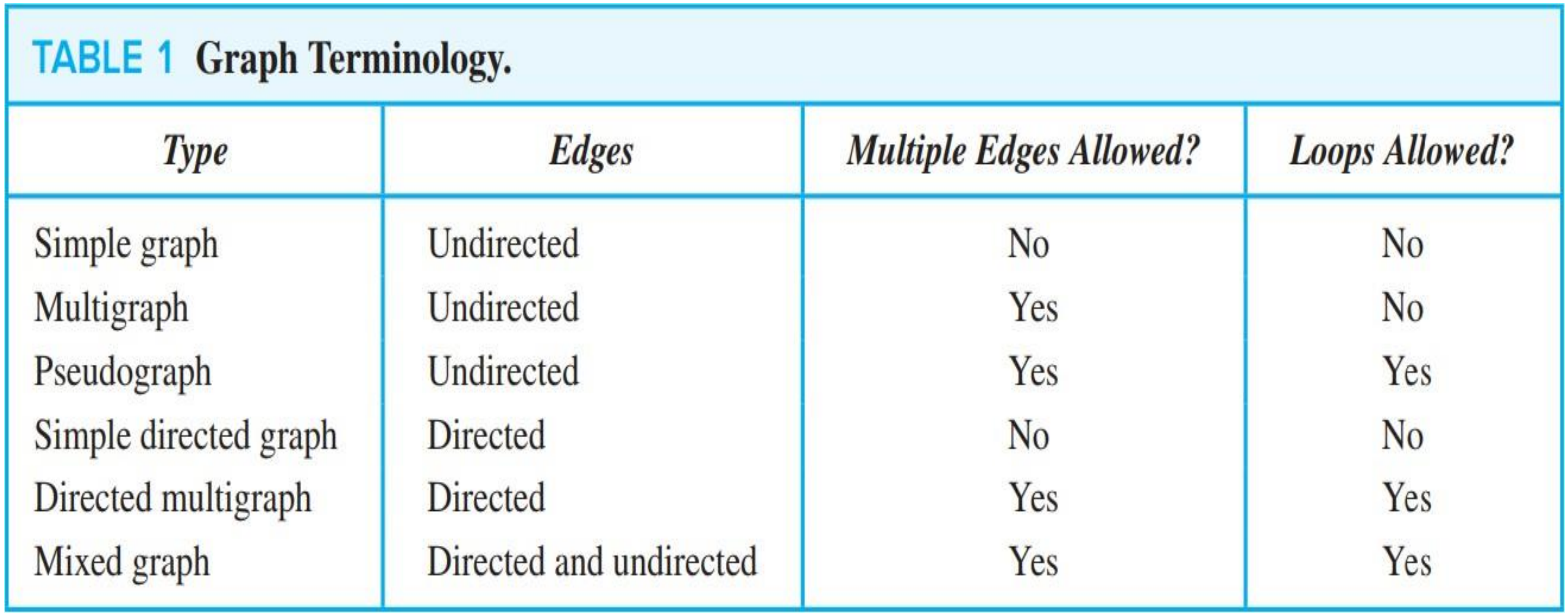


TABLE 1 **Graph Terminology.**

Type	*Edges*	*Multiple Edges Allowed?*	*Loops Allowed?*
Simple graph	Undirected	No	No
Multigraph	Undirected	Yes	No
Pseudograph	Undirected	Yes	Yes
Simple directed graph	Directed	No	No
Directed multigraph	Directed	Yes	Yes
Mixed graph	Directed and undirected	Yes	Yes

Graphs and Graph Models (13/29)

Structure of A Graph:

Three key questions can help us understand the structure of a graph:

1. Are the edges of the graph undirected or directed (or both)?
2. If the graph is undirected, are multiple edges present that connect the same pair of vertices? If the graph is directed, are multiple directed edges present?
3. Are loops present?

Graphs and Graph Models (14/29)

Graph Models:

Graphs are used in a wide variety of models.

- Social Networks.
- Communication Networks.
- Information Networks.
- Transportation Networks.
- Biological Networks.
- Software Design Applications.
- Tournaments.
- Others…

Graphs and Graph Models (15/29)

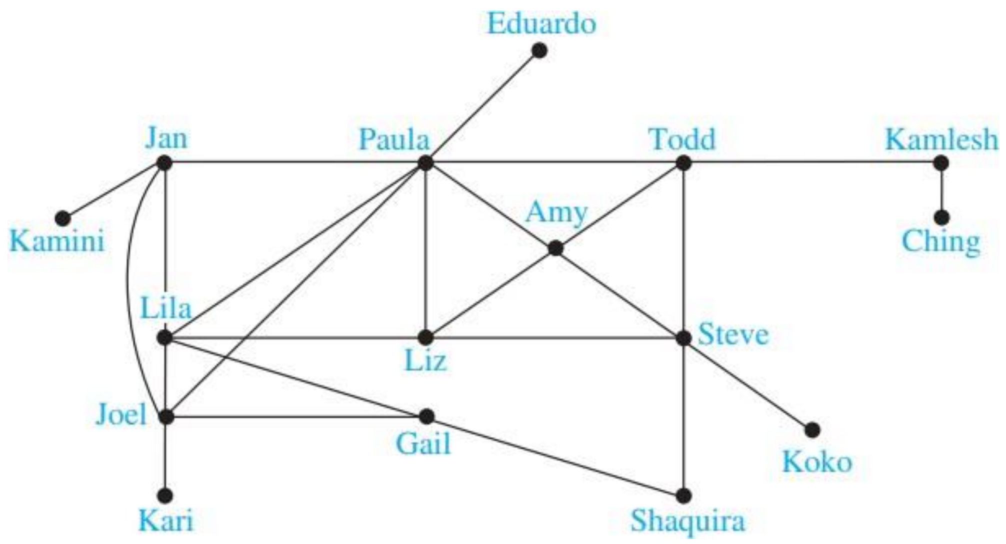
Social Networks:

Graphs are extensively used to model social structures based on different kinds of relationships between people or groups of people.

Graphs and Graph Models (16/29)

Social Networks:

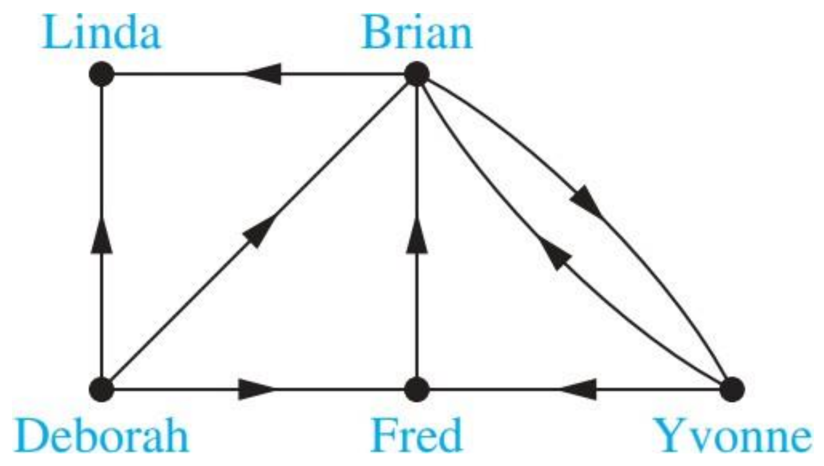
Acquaintanceship and Friendship Graphs: We can use a simple graph to represent whether two people know each other, that is, whether they are acquainted, or whether they are friends (either in the real world or in the virtual world via a social networking site such as Facebook).



Graphs and Graph Models (17/29)

Social Networks:

Influence Graphs: In studies of group behavior, it is observed that certain people can influence the thinking of others.



Graphs and Graph Models (18/29)

Social Networks:

Collaboration Graphs: is used to model social networks where two people are related by working together in a particular way.

- The Hollywood Links graph is a collaborator graph that represents actors by vertices and connects two actors with an edge if they have worked together on a movie or television show. The Hollywood graph is a huge graph with more than 2.9 million vertices (as of early 2018).

Graphs and Graph Models (19/29)

Social Networks:

Collaboration Graphs: is used to model social networks where two people are related by working together in a particular way.

- In an academic collaboration graph, vertices represent people and edges link two people if they have jointly published a paper. The collaboration graph for people who have published research papers in mathematics was found in 2004 to have more than 400,000 vertices and 675,000 edges.

Graphs and Graph Models (20/29)

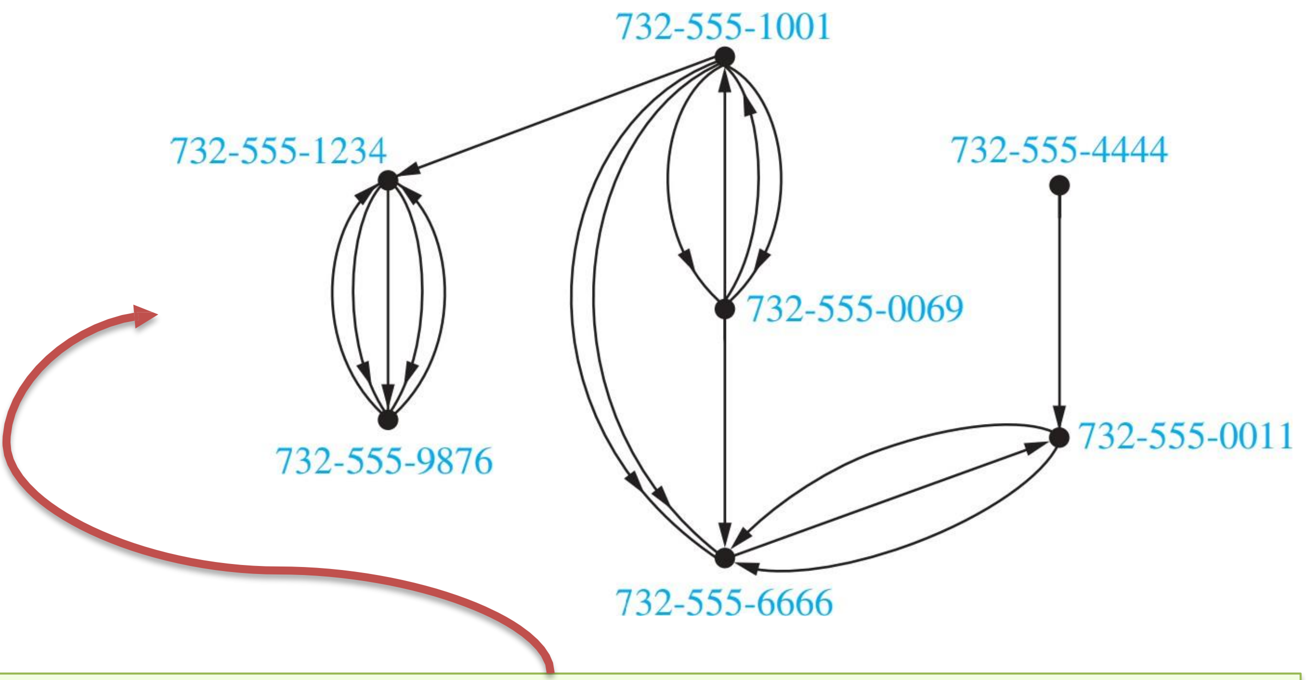
Communication Networks:

We can model different communications networks using vertices to represent devices and edges to represent the particular type of communications links of interest.

Graphs and Graph Models (21/29)

Communication Networks:

- **Call Graphs**: Graphs can be used to model telephone calls made in a network, such as a long-distance telephone network.

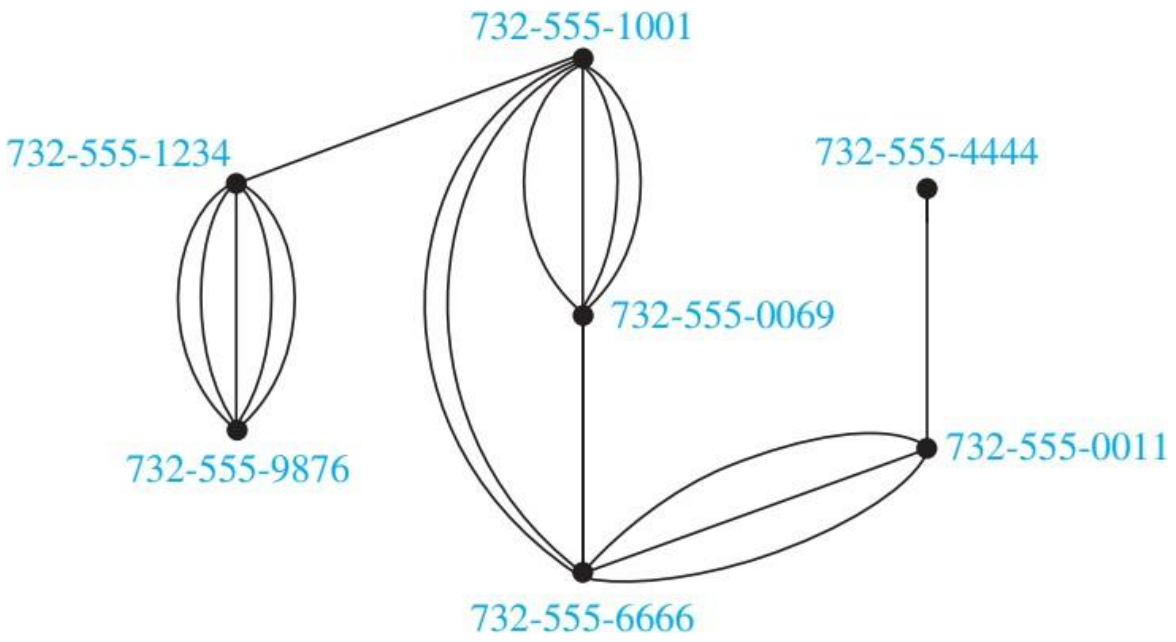


Three calls have been made from 732-555-1234 to 732-555-9876

Graphs and Graph Models (22/29)

Communication Networks:

- **Call Graphs**: Graphs can be used to model telephone calls made in a network, such as a long-distance telephone network.

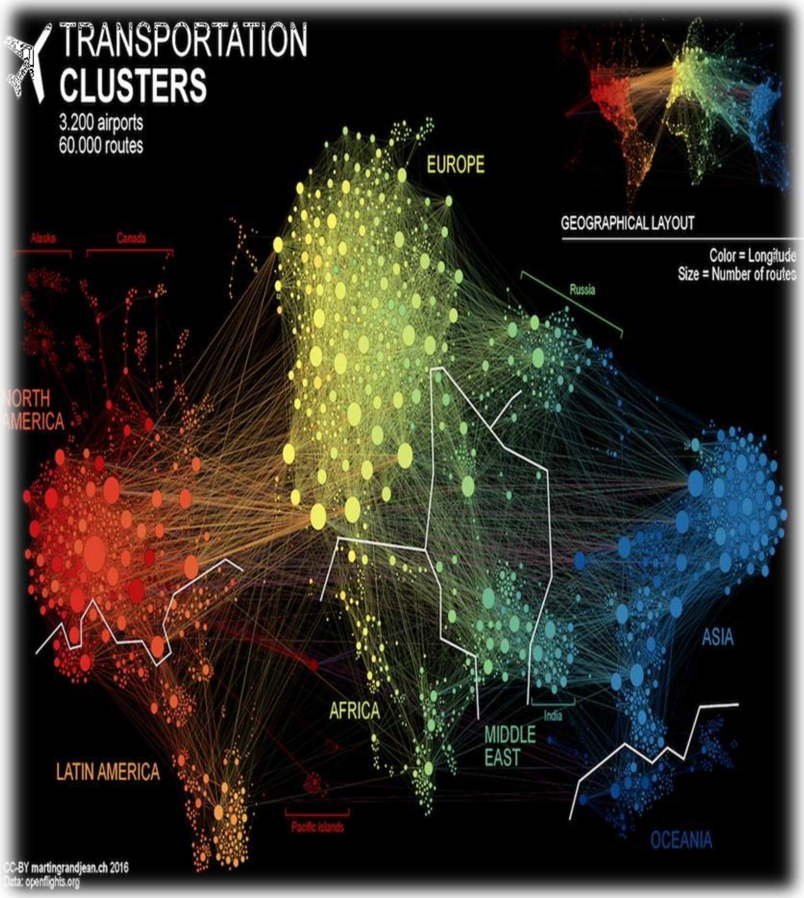


When we care only whether there has been a call connecting two telephone numbers

Graphs and Graph Models (23/29)

Transportation Networks:

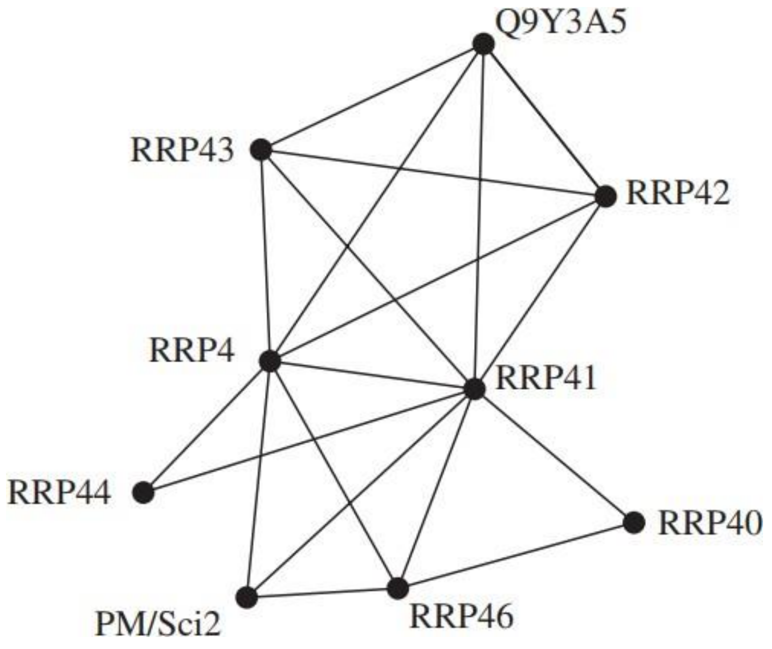
We can use graphs to model many different types of transportation networks, including ***road***, ***air***, and ***rail*** networks, as well as ***shipping*** networks.



Graphs and Graph Models (24/29)

Biological Networks:

Many aspects of the biological sciences can be modeled using graphs. ***Protein Interaction Graphs***: A protein interaction in a living cell occurs when two or more proteins in that cell bind to perform a biological function.



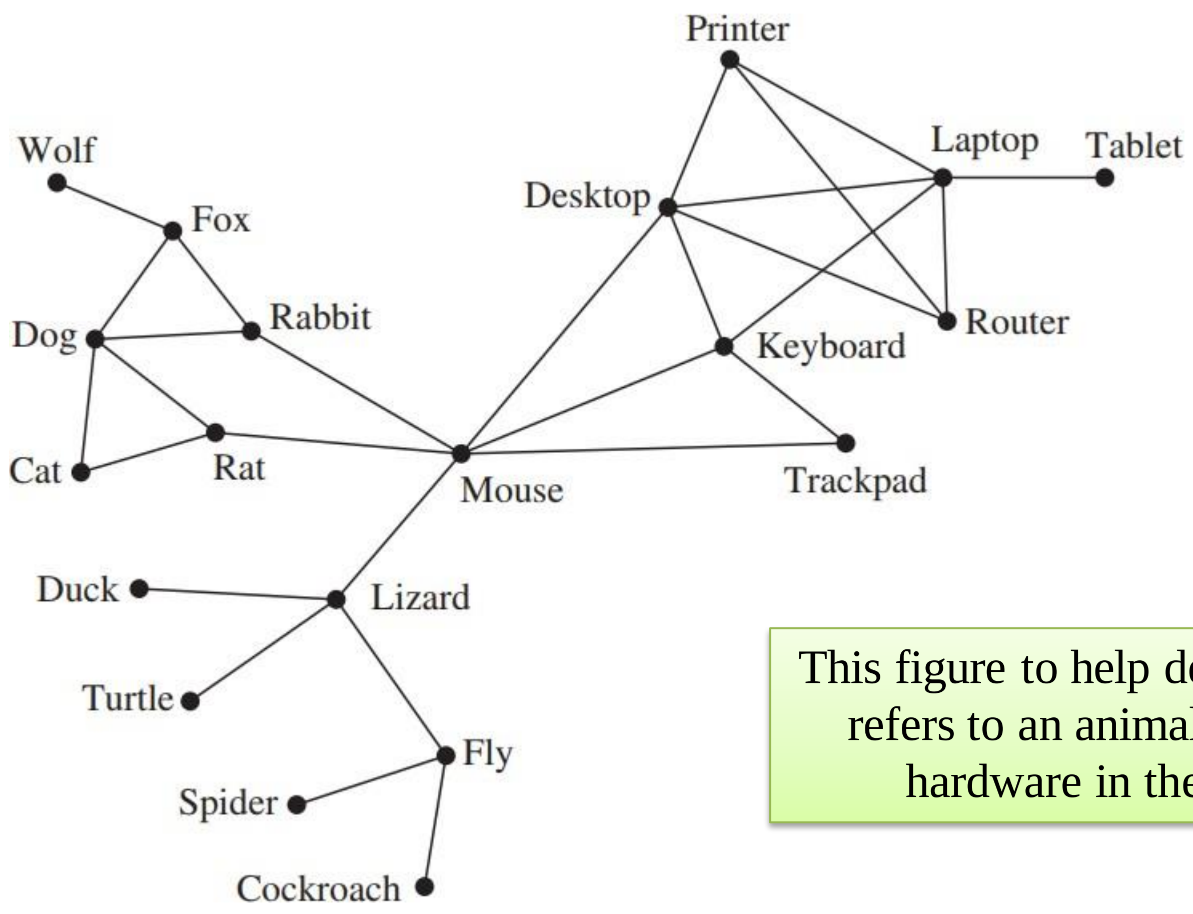
Graphs and Graph Models (25/29)

Semantic Networks:

Graph models are used extensively in natural language understanding and in information retrieval. ***Natural language understanding (NLU)*** is the subject of enabling machines to disassemble and parse human speech. Its goal is to allow machines to understand and communicate as humans do.

Graphs and Graph Models (26/29)

Semantic Networks:



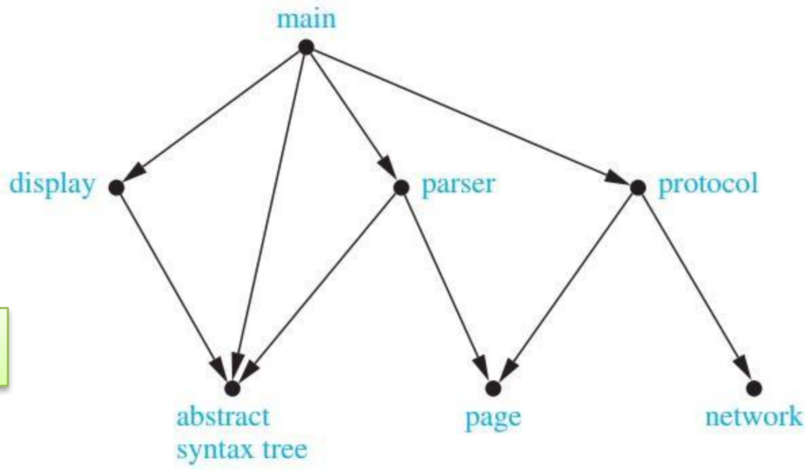
This figure to help determine ***mouse*** refers to an animal or computer hardware in the sentence.

Graphs and Graph Models (27/29)

Software Design Applications:

Graph models are useful tools in the design of software. ***Module Dependency Graphs.*** One of the most important tasks in designing software is how to structure a program into different parts, or modules. Understanding how the different modules of a program interact is essential not only for program design, but also for testing and maintenance of the resulting software.

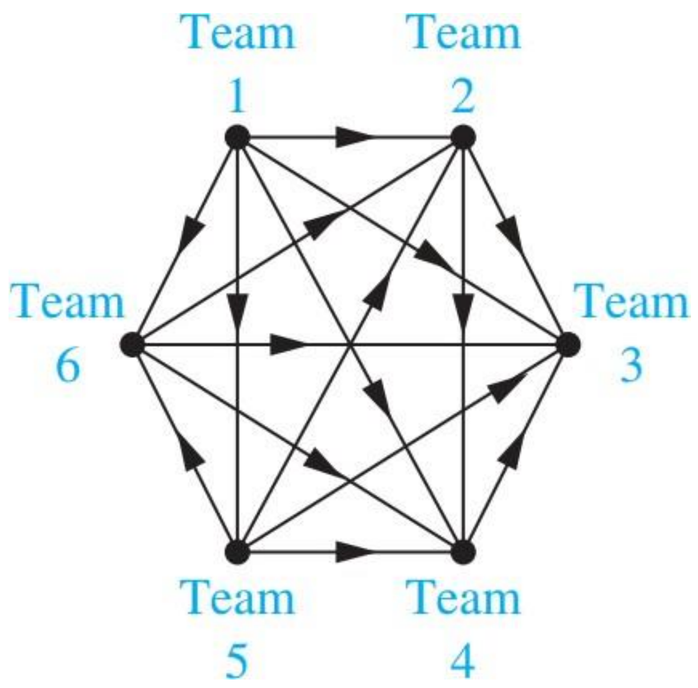
web browser



Graphs and Graph Models (28/29)

Tournaments:

Round-Robin Tournaments. A tournament where each team plays every other team exactly once and no draws are allowed.

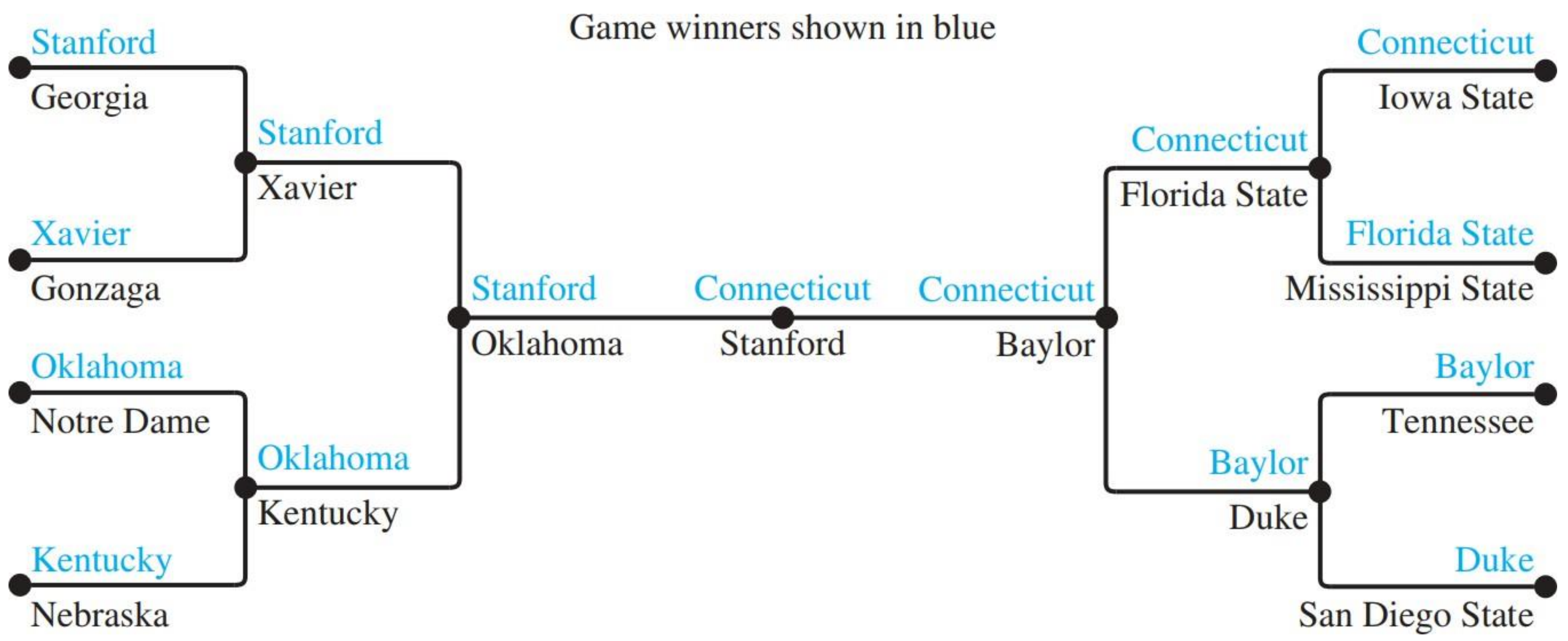


We see that Team 1 is undefeated in this tournament, and Team 3 is winless.

Graphs and Graph Models (29/29)

Tournaments:

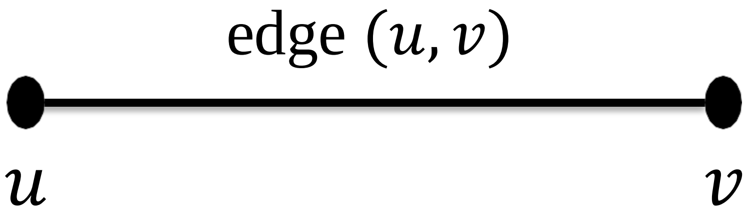
Single-Elimination Tournaments. A tournament where each contestant is eliminated after one loss



Basic Graph Terminology (1/25)

Definition 1:

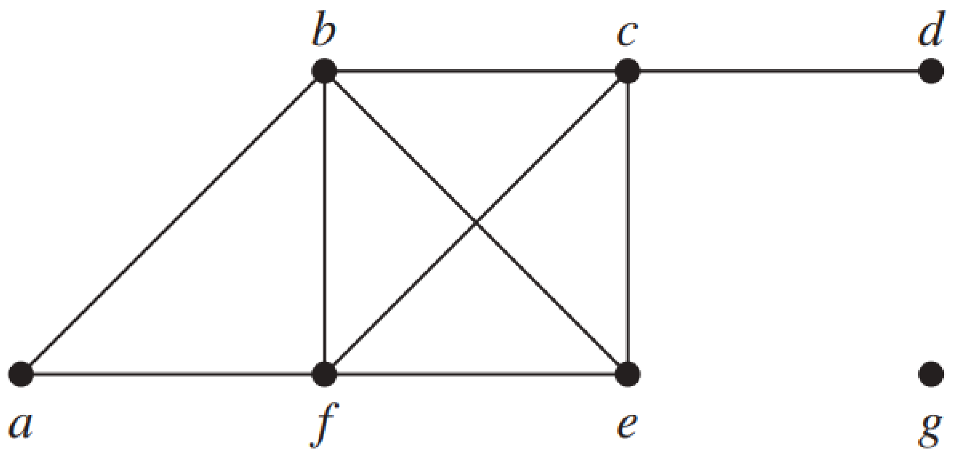
Two vertices u and v in an undirected graph G are called **adjacent** (or **neighbors**) in G if u and v are endpoints of an edge e of G. Such an edge e is called *incident with* the vertices u and v and e is said to *connect* u and v.



Basic Graph Terminology (2/25)

Definition 2:

The set of all neighbors of a vertex v of $G = (V, E)$, denoted by $N(v)$, is called the neighborhood of v. If A is a subset of V, we denote by $N(A)$ the set of all vertices in G that are adjacent to at least one vertex in A. So, $N(A) = \sqcup_{v \in A} N(v)$.



$\boldsymbol{N(a) = \{b, f\}}$

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

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

$\boldsymbol{N(d) = \{c\}}$

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

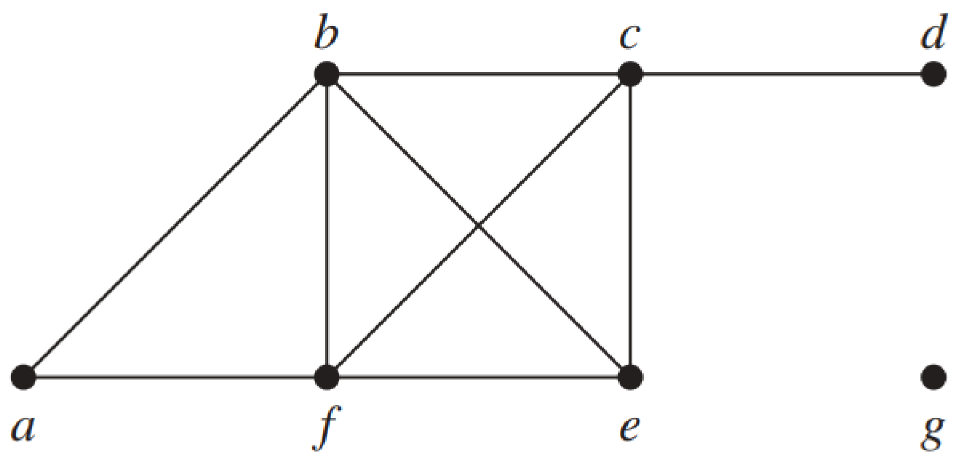
$\boldsymbol{N(f) = \{a, b, c, e\}}$

$\boldsymbol{N(\mathrm{g}) = \emptyset}$

Basic Graph Terminology (3/25)

Definition 3:

The **degree** of a **vertex** in an *undirected* graph is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex. The degree of the vertex v is denoted by deg(v).



$\deg(a) = 2$

$\deg(b) = 4$

$\deg(c) = 4$

$\deg(d) = 1$

$\deg(e) = 3$

$\deg(f) = 4$

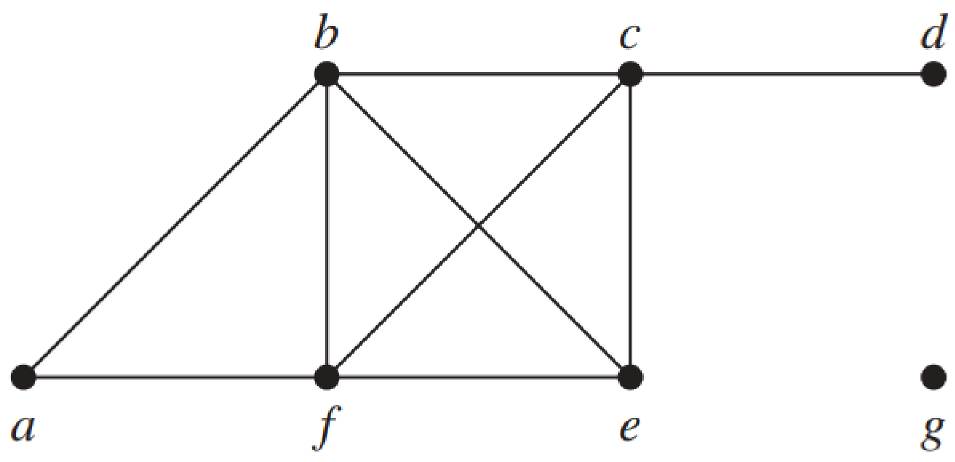
$\deg(g) = 0$

Basic Graph Terminology (4/25)

Isolated:

A vertex of degree zero is called **isolated**. It follows that an isolated vertex is not adjacent to any vertex.

Vertex g is ***isolated***.



$\deg(a) = 2$

$\deg(b) = 4$

$\deg(c) = 4$

$\deg(d) = 1$

$\deg(e) = 3$

$\deg(f) = 4$

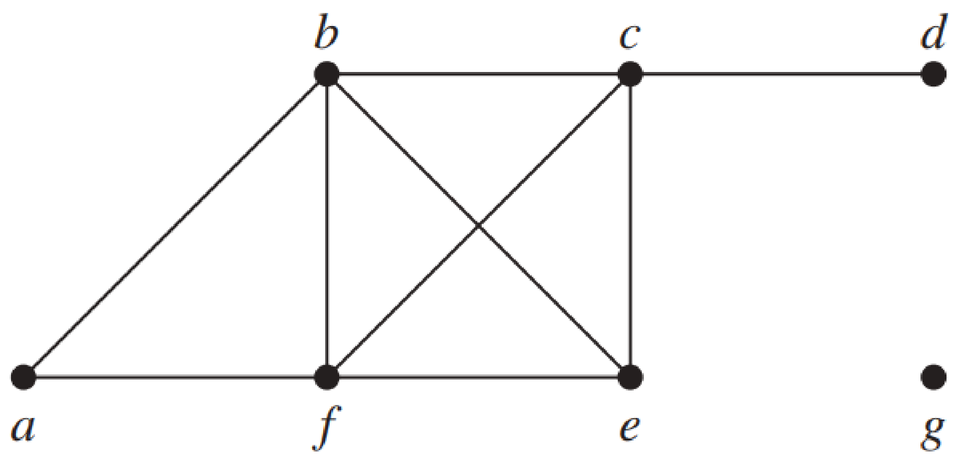
$\boxed{\deg(\mathrm{g}) = 0}$

Basic Graph Terminology (5/25)

Pendant:

A vertex is **pendant** if and only if it has degree *one*.

Vertex d is ***pendant***.



$\deg(a) = 2$

$\deg(b) = 4$

$\deg(c) = 4$

$\boxed{\deg(d) = 1}$

$\deg(e) = 3$

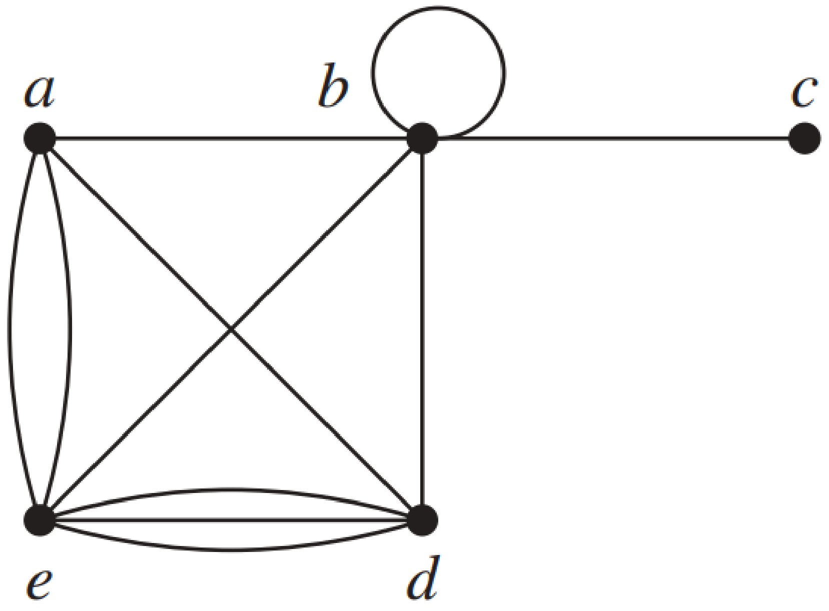
$\deg(f) = 4$

$\deg(g) = 0$

Basic Graph Terminology (6/25)

Example 1:

What are the degrees and what are the neighborhoods of the vertices in the following graph?



$\deg(a) =$

$\deg(b) =$

$\deg(c) =$

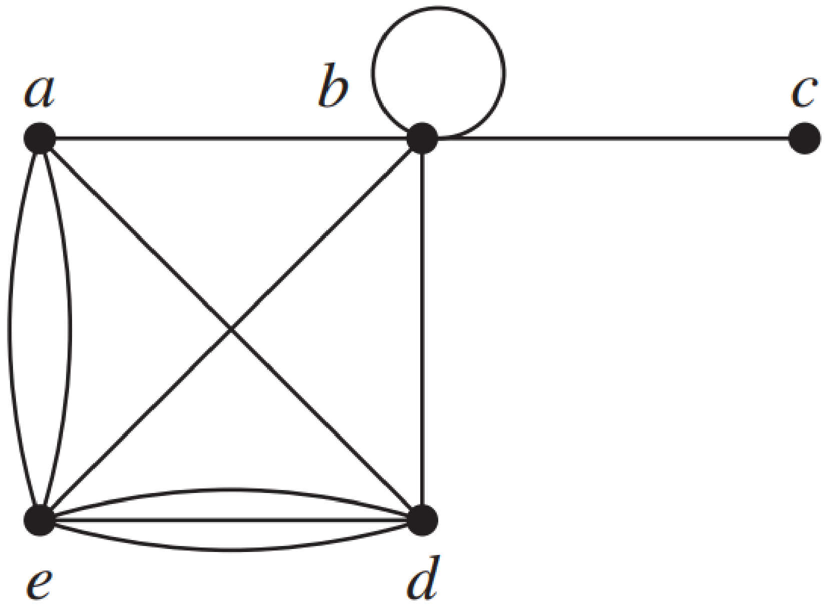
$\deg(d) =$

$\deg(e) =$

Basic Graph Terminology (7/25)

Example 1:

What are the degrees and what are the neighborhoods of the vertices in the following graph?



$\deg(a) = 4$

$\deg(b) = 6$

$\deg(c) = 1$

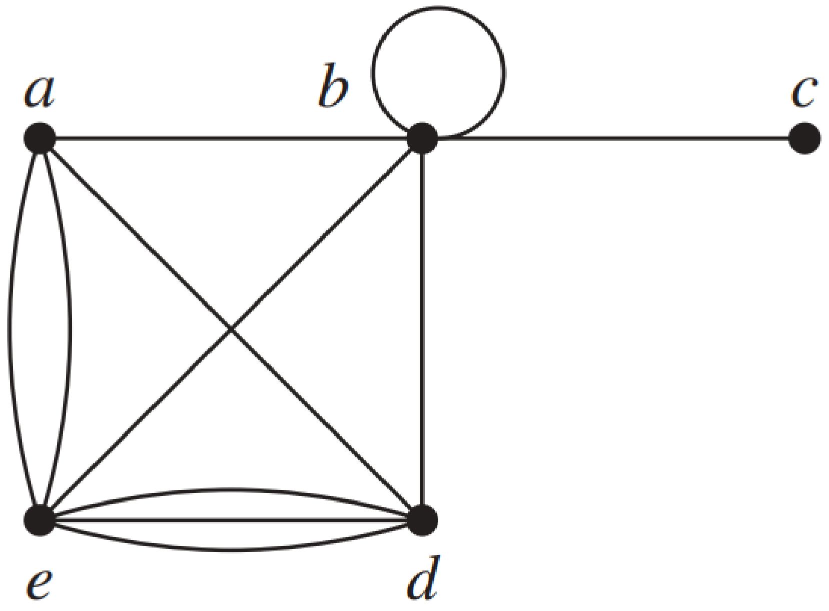
$\deg(d) = 5$

$\deg(e) = 6$

Basic Graph Terminology (8/25)

Example 1:

What are the degrees and what are the neighborhoods of the vertices in the following graph?



$\deg(a) = 4$

$\deg(b) = 6$

$\boxed{\deg(c) = 1}$

$\deg(d) = 5$

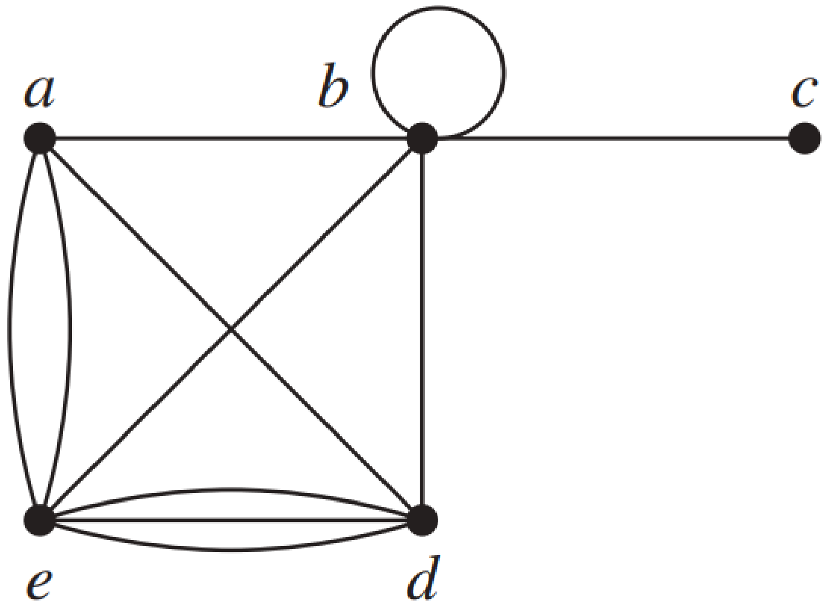
$\deg(e) = 6$

Vertex c is pendant

Basic Graph Terminology (9/25)

Example 1:

What are the degrees and what are the neighborhoods of the vertices in the following graph?



$N(a) =$

$N(b) =$

$N(c) =$

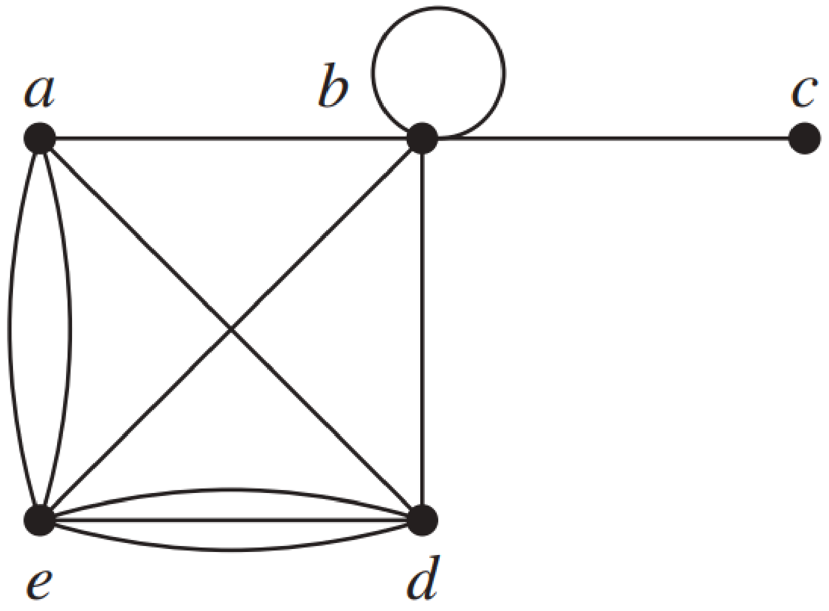
$N(d) =$

$N(e) =$

Basic Graph Terminology (10/25)

Example 1:

What are the degrees and what are the neighborhoods of the vertices in the following graph?



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

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

$N(c) = \{b\}$

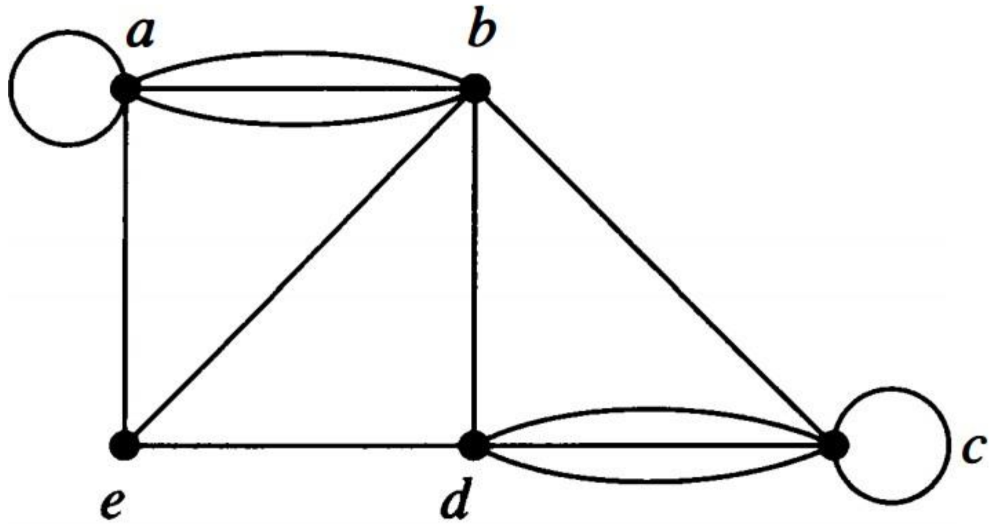
$N(d) = \{a, b, e\}$

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

Basic Graph Terminology (11/25)

Example 2:

What are the degrees and what are the neighborhoods of the vertices in the following graph?



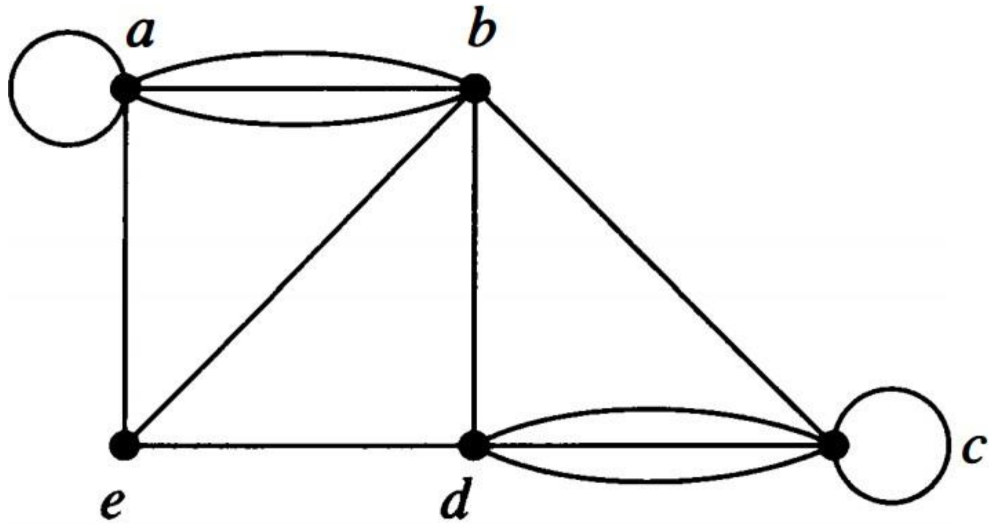
Number of vertices = 5

Number of edges = 13

Basic Graph Terminology (12/25)

Example 2:

What are the degrees and what are the neighborhoods of the vertices in the following graph?



$\deg(a) =$

$\deg(b) =$

$\deg(c) =$

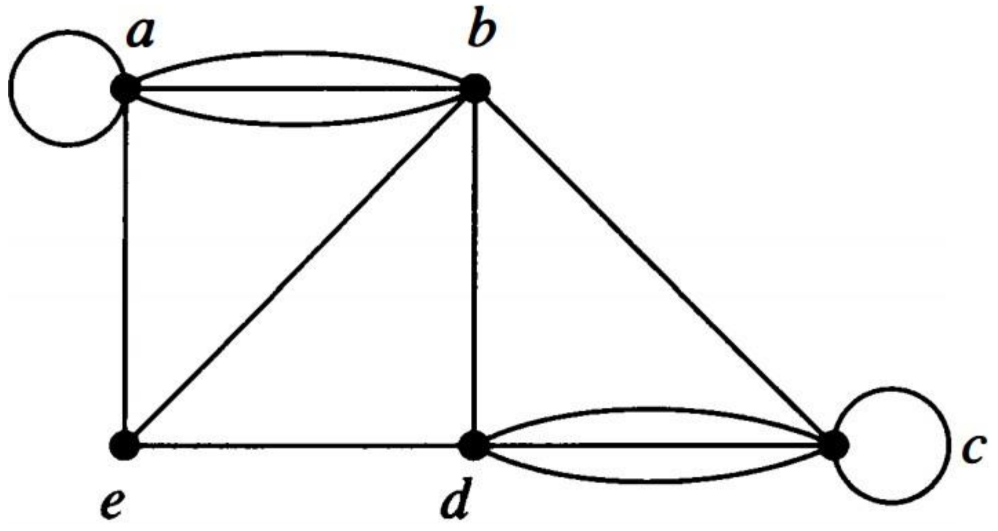
$\deg(d) =$

$\deg(e) =$

Basic Graph Terminology (13/25)

Example 2:

What are the degrees and what are the neighborhoods of the vertices in the following graph?



$\deg(a) = 6$

$\deg(b) = 6$

$\deg(c) = 6$

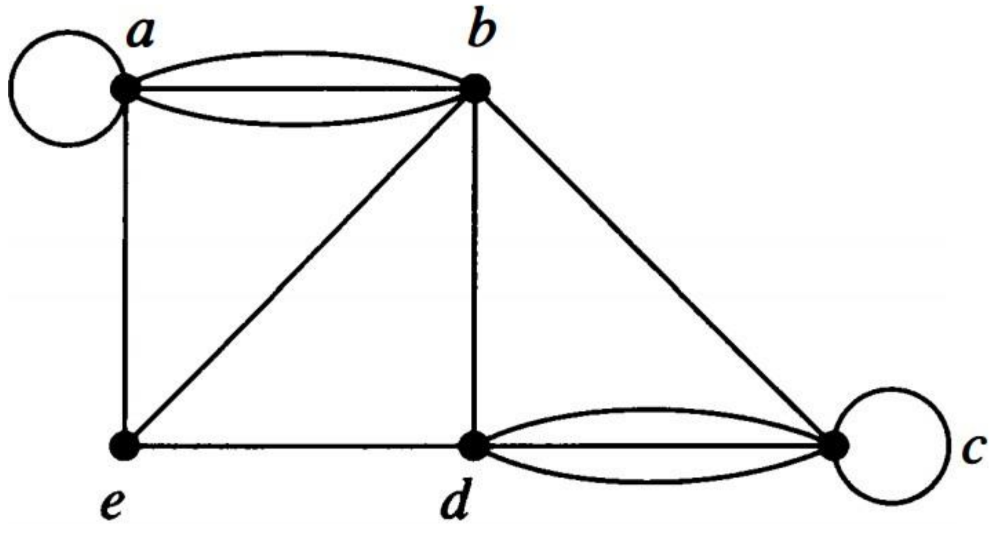
$\deg(d) = 5$

$\deg(e) = 3$

Basic Graph Terminology (14/25)

Example 2:

What are the degrees and what are the neighborhoods of the vertices in the following graph?



$N(a) =$

$N(b) =$

$N(c) =$

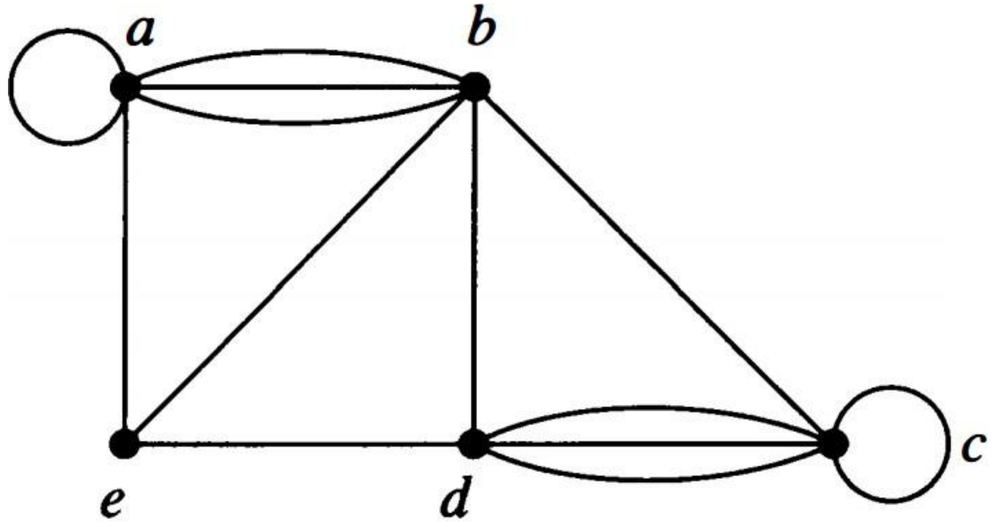
$N(d) =$

$N(e) =$

Basic Graph Terminology (15/25)

Example 2:

What are the degrees and what are the neighborhoods of the vertices in the following graph?



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

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

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

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

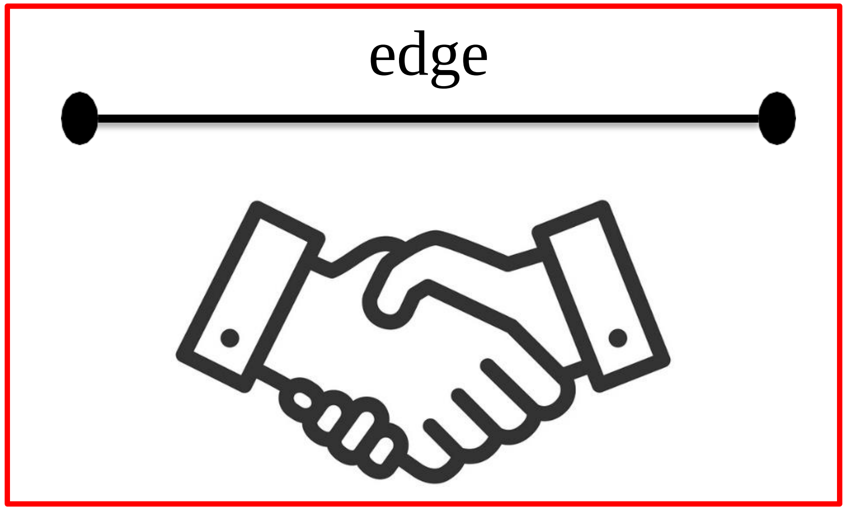
$N(e) = \{a, b, d\}$

Basic Graph Terminology (16/25)

The Handshaking Theorem:

Let $G = (V, E)$ be undirected graph with m edges. Then

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



Edge having two endpoints and a handshake involving two hands.

Basic Graph Terminology (17/25)

Example 3:

How many edges are there in an undirected graph with 10 vertices each of degree six?

Basic Graph Terminology (18/25)

Example 3: Answer

How many edges are there in a graph with 10 vertices each of degree six?

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

Solution:

Because the sum of the degrees of the vertices is $6 \cdot 10 = 60$, it follows that $2m = 60$. Therefore, $m = 30$.

Basic Graph Terminology (19/25)

Theorem 2:

An undirected graph has an even number of vertices of odd degree.

<u>Proof:</u>

Let V_1 and V_2 be the set of vertices of even degree and the set of vertices of odd degree, respectively, in an undirected graph $G = (V, E)$ with m edges.

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

∵even

∵even

Basic Graph Terminology (19/25)

Theorem 2:

An undirected graph has an even number of vertices of odd degree.

Proof:

Let V_1 and V_2 be the set of vertices of even degree and the set of vertices of odd degree, respectively, in an undirected graph $G = (V, E)$ with m edges.

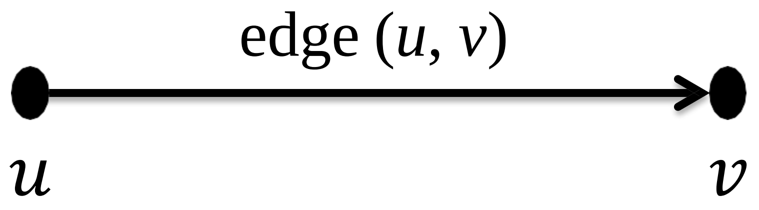
$$2m = \sum_{v \in V} \deg(v) = \sum_{v \in V_1} \deg(v) + \sum_{v \in V_2} \deg(v)$$

∵even ∵even ∴even

Basic Graph Terminology (20/25)

Definition 4:

When (u, v) is an edge of the graph G with *directed* edges, u is said to be ***adjacent to*** v and v is said to be ***adjacent from*** u. The vertex u is called the **initial vertex** of (u, v), and v is called the **terminal** or **end vertex** of (u, v). The initial vertex and terminal vertex of a loop are the same.



Basic Graph Terminology (21/25)

Definition 5:

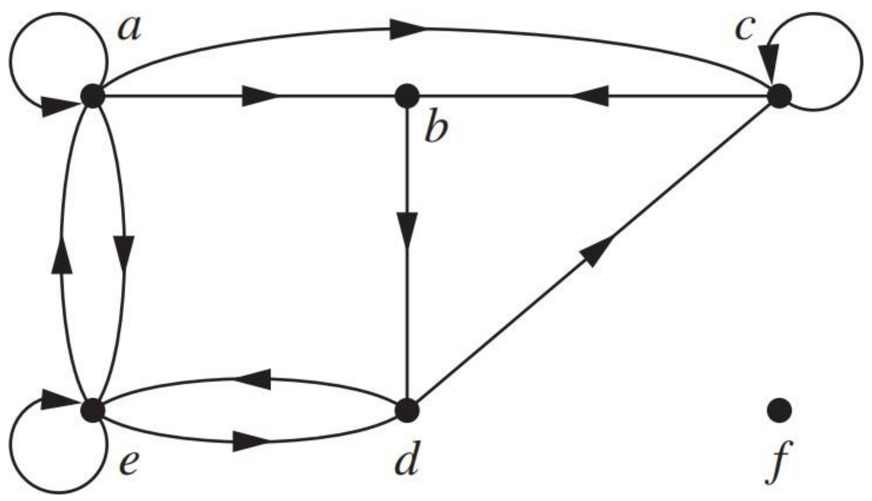
In a graph with directed edges the **in-degree** of a vertex v, denoted by $\deg^{-}(v)$, is the number of edges with v as their terminal vertex.

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

(Note that a loop at a vertex contributes 1 to both the in-degree and the out-degree of this vertex.)

Basic Graph Terminology (22/25)

Example 4:



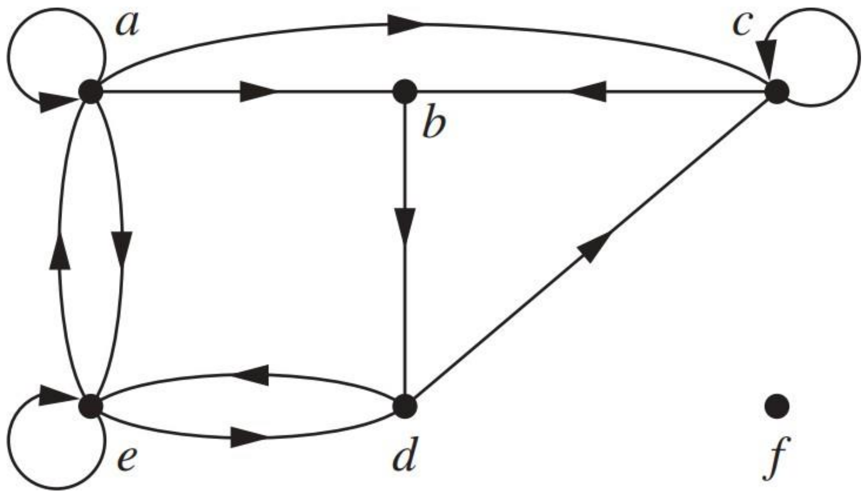
Number of vertices =

Number of edges =

$\deg^-(a) =$	$\deg^+(a) =$
$\deg^-(b) =$	$\deg^+(b) =$
$\deg^-(c) =$	$\deg^+(c) =$
$\deg^-(d) =$	$\deg^+(d) =$
$\deg^-(e) =$	$\deg^+(e) =$
$\deg^-(f) =$	$\deg^+(f) =$

Basic Graph Terminology (23/25)

Example 4:



Number of vertices = 6

Number of edges = 12

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

Basic Graph Terminology (24/25)

Theorem 3:

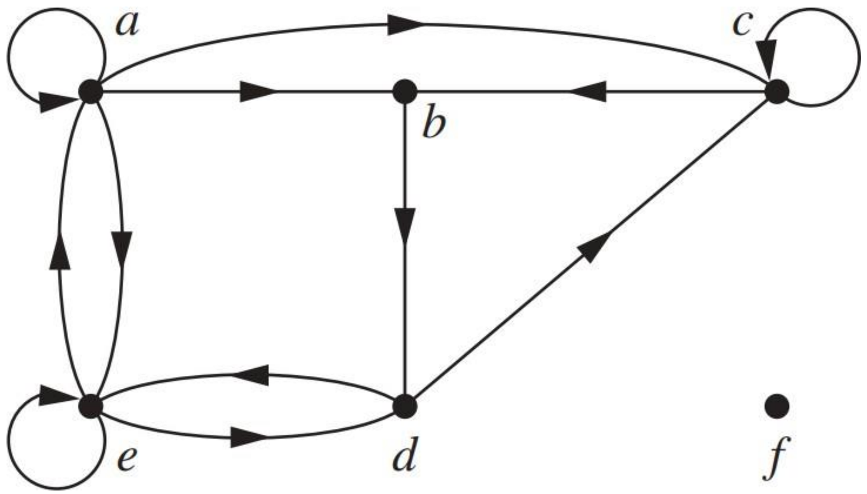
Let $G = (V, E)$ be a graph with directed edges. Then

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

Basic Graph Terminology (25/25)

Example 4:

Recall:



Number of vertices = 6

Number of edges = 12

$\deg^-(a) = 2$

$\deg^-(b) = 2$

$\deg^-(c) = 3$

$\deg^-(d) = 2$

$\deg^-(e) = 3$

$\deg^-(f) = 0$

$\deg^+(a) = 4$

$\deg^+(b) = 1$

$\deg^+(c) = 2$

$\deg^+(d) = 2$

$\deg^+(e) = 3$

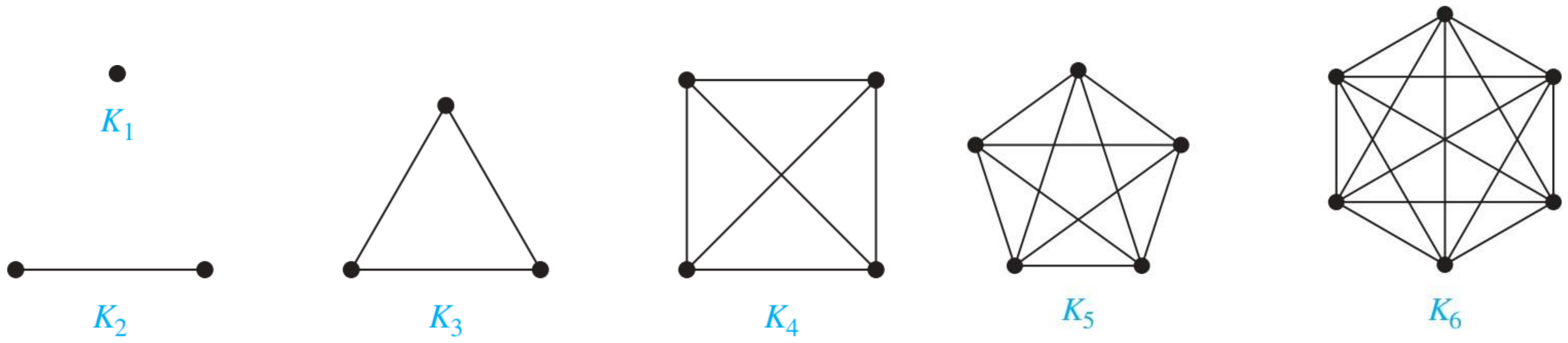
$\deg^+(f) = 0$

Some Special Simple Graphs (1/16)

Complete Graphs:

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

The graphs K_n, for n = 1, 2, 3, 4, 5, 6, are:

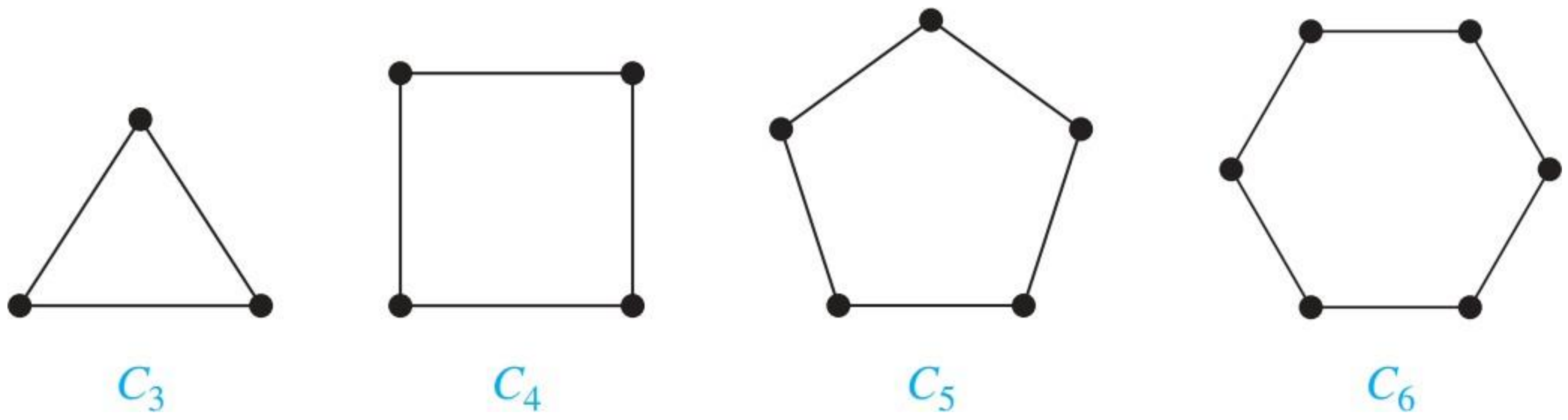


Some Special Simple Graphs (2/16)

Cycles:

The cycle C_n, $n \geq 3$, consists of n vertices $v_1, v_2, \ldots, v_n$ and edges $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}$ and $\{v_n, v_1\}$.

The cycles C_3, C_4, C_5, and C_6 are:

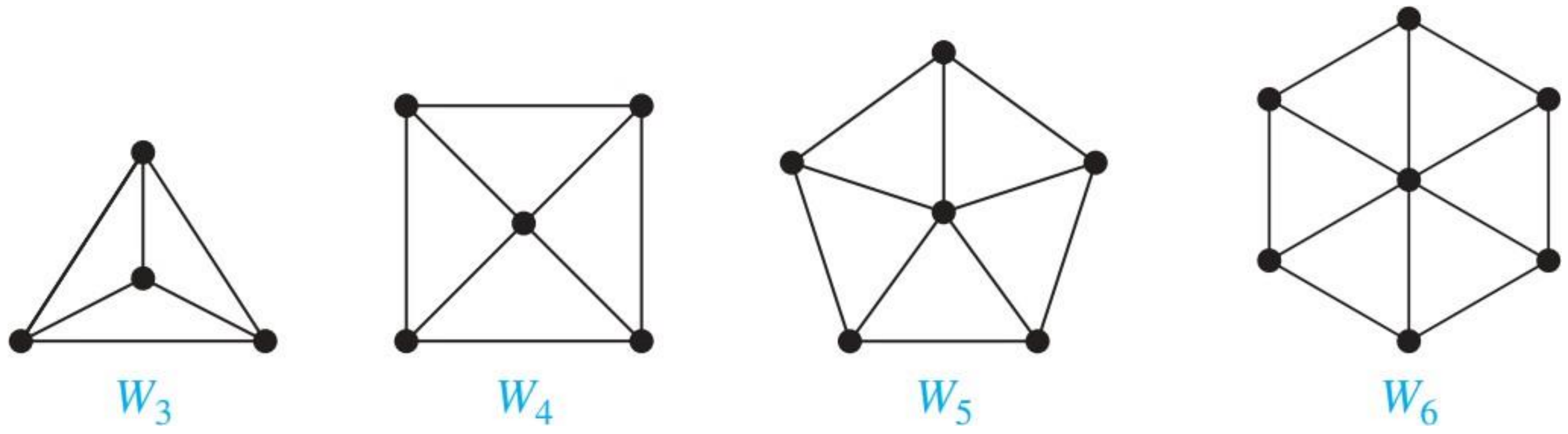


Some Special Simple Graphs (3/16)

Wheels:

We obtain the wheel W_n when we add an additional vertex to the cycle C_n, for $n \geq 3$, and connect this new vertex to each of the n vertices in C_n, by new edges.

The wheels W_3, W_4, W_5, and W_6 are:

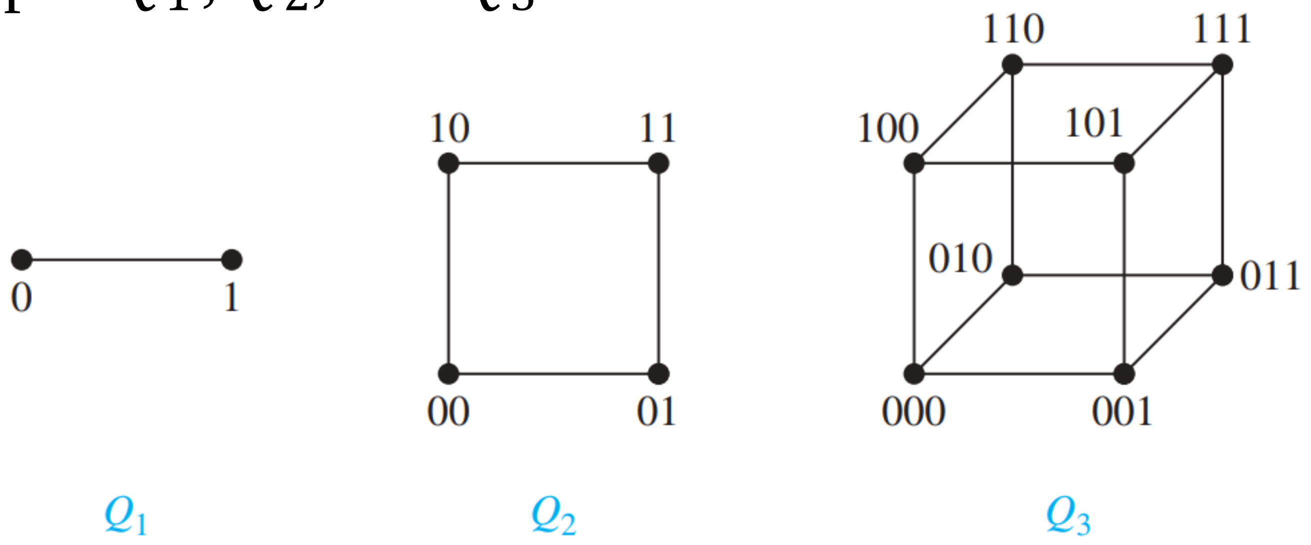


Some Special Simple Graphs (4/16)

n-Cubes:

The n-dimensional hypercube, or n-cube, denoted by Q_n, is the graph that has vertices representing the 2^n bit strings of length n. Two vertices are adjacent if and only if the bit strings that they represent differ in exactly one-bit position.

The graphs Q_1, Q_2, and Q_3 are:



Some Special Simple Graphs (5/16)

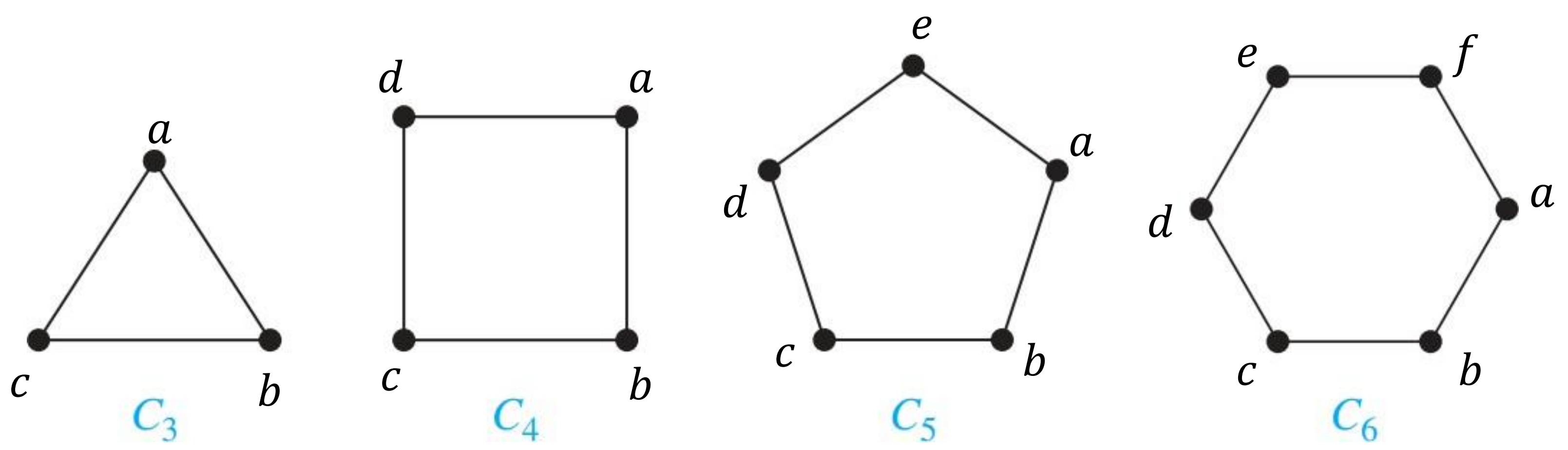
Definition 6: Bipartite Graphs:

A simple graph G is called **bipartite** if its vertex set V can be partitioned into two disjoint sets V_1 ad V_2 such that every edge in the graph connects a vertex in V_1 and a vertex in V_2 (so that no edge in G connects either two vertices in V_1 or two vertices in V_2). When this condition holds, we call the pair (V_1, V_2) a bipartition of the vertex set V of G.

Some Special Simple Graphs (6/16)

Example 1:

Determine whether the graph is bipartite or not?

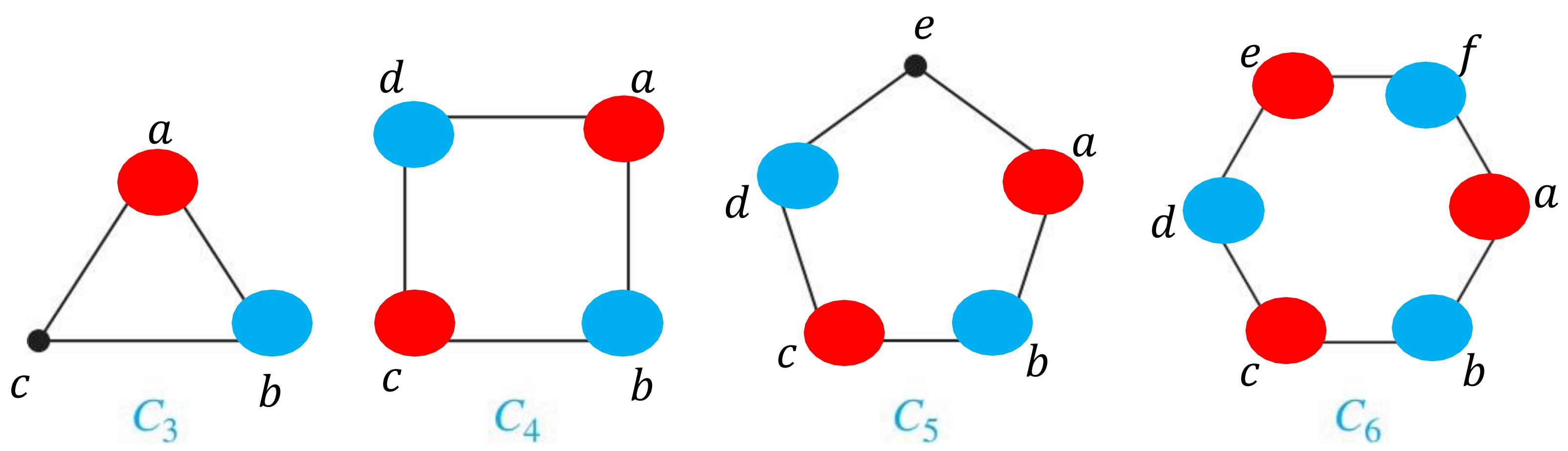


Determining whether it is possible to assign either **red** or **blue** to each vertex so that *no two adjacent vertices are assigned the same color*.

Some Special Simple Graphs (6/16)

Example 1: Answer

Determine whether the graph is bipartite or not?

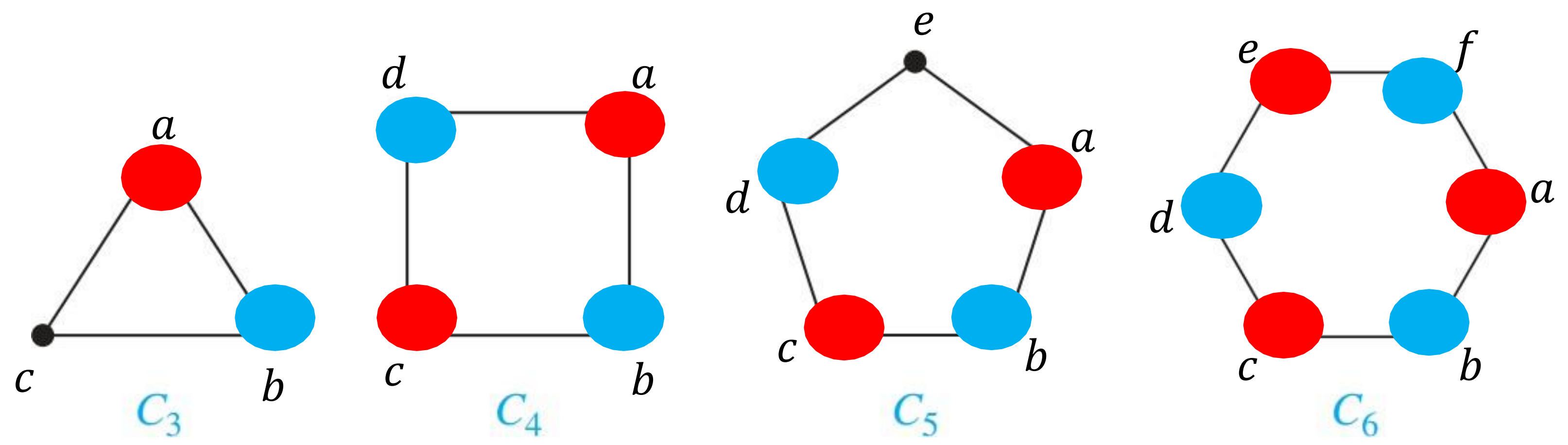


Determining whether it is possible to assign either **red** or **blue** to each vertex so that *no two adjacent vertices are assigned the same color*.

Some Special Simple Graphs (6/16)

Example 1: Answer

Determine whether the graph is bipartite or not?



bipartite

$V_1 = \{a, c\}$
$V_2 = \{b, d\}$
every edge of C_4 connects a vertex in V_1 and a vertex in V_2.

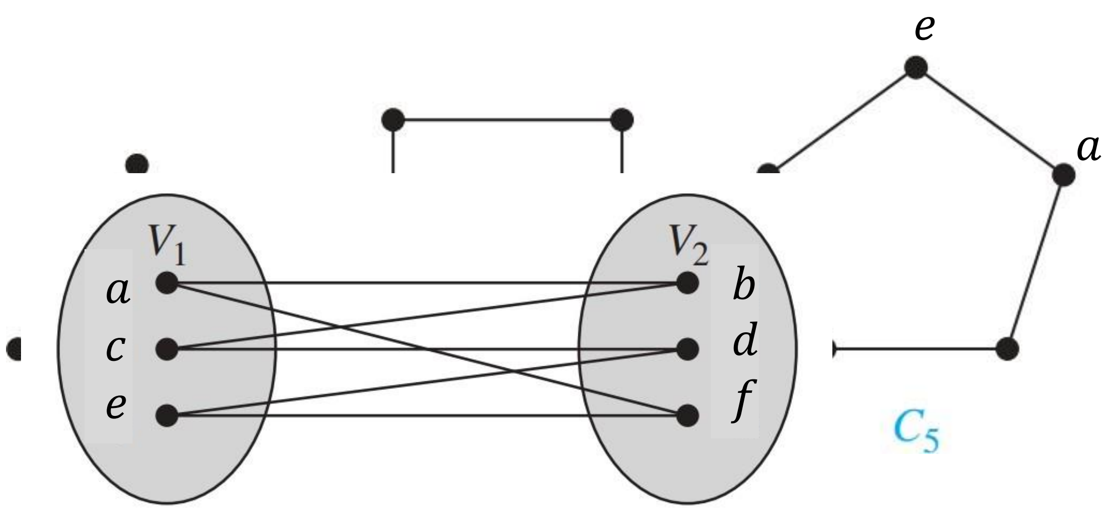
bipartite

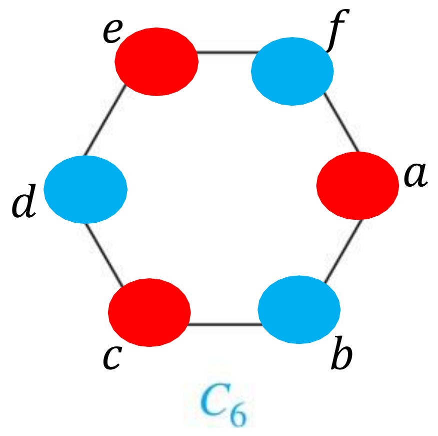
$V_1 = \{a, c, e\}$
$V_2 = \{b, d, f\}$
every edge of C_6 connects a vertex in V_1 and a vertex in V_2.

Some Special Simple Graphs (6/16)

Example 1: Answer

Determine whether the graph is bipartite or not?





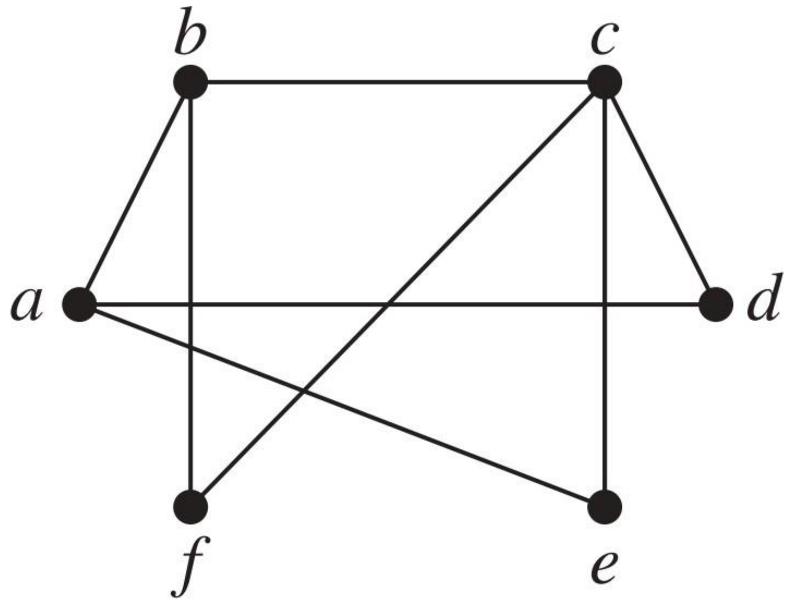
bipartite

$V_1 = \{a, c, e\}$
$V_2 = \{b, d, f\}$
every edge of C_6 connects a vertex in V_1 and a vertex in V_2.

Some Special Simple Graphs (7/16)

Example 2:

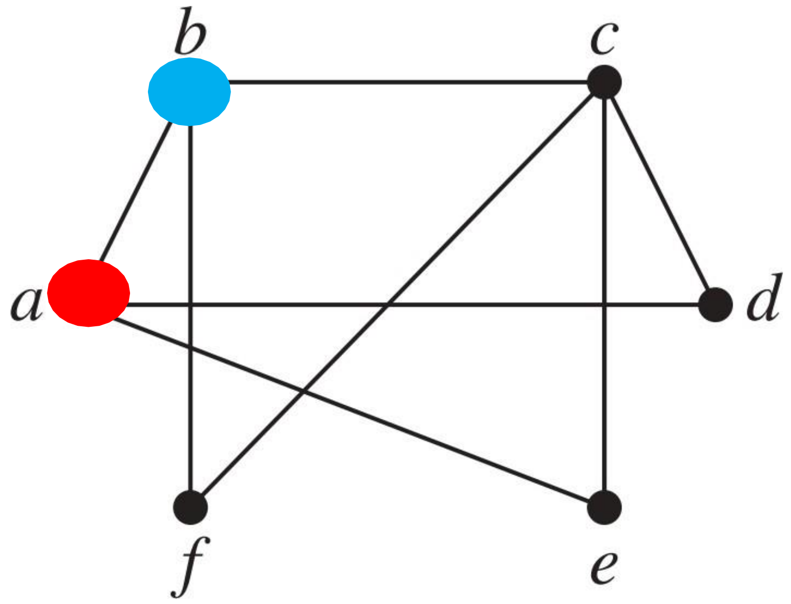
Determine whether the graph is bipartite or not?



Some Special Simple Graphs (7/16)

Example 2: Answer

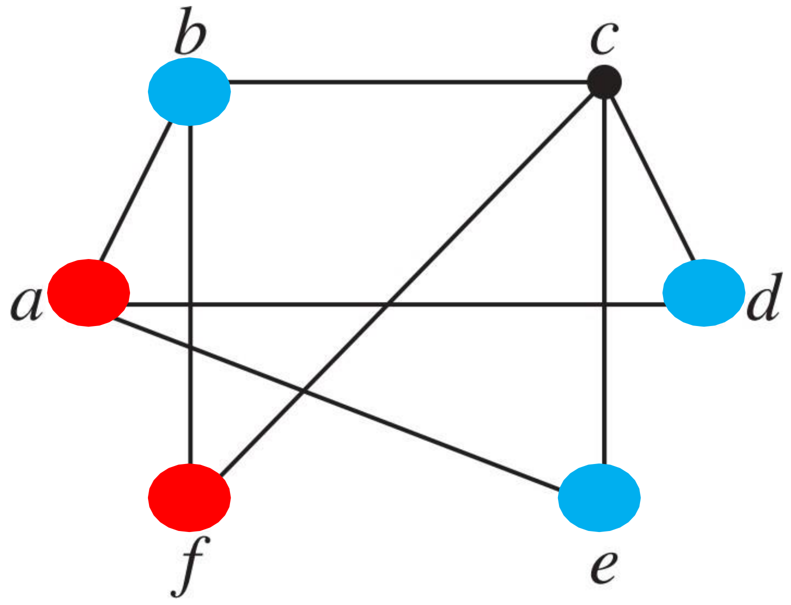
Determine whether the graph is bipartite or not?



Some Special Simple Graphs (7/16)

Example 2: Answer

Determine whether the graph is bipartite or not?

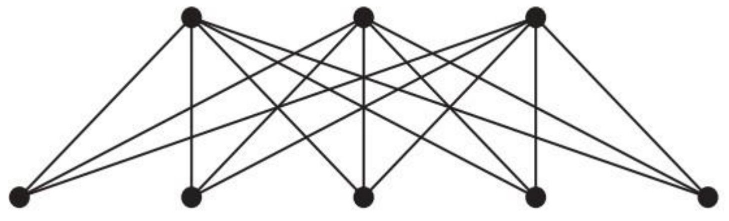


Not bipartite

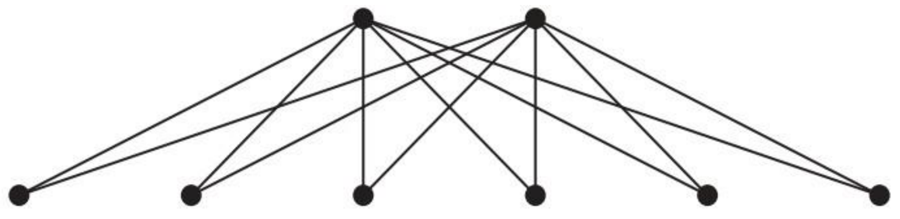
Some Special Simple Graphs (8/16)

Complete Bipartite Graphs:

A complete bipartite graph $K_{m,n}$ is a graph that has its vertex set partitioned into two subsets of m and n vertices, respectively with an edge between two vertices if and only if one vertex is in the first subset and the other vertex is in the second subset.



$K_{3,5}$

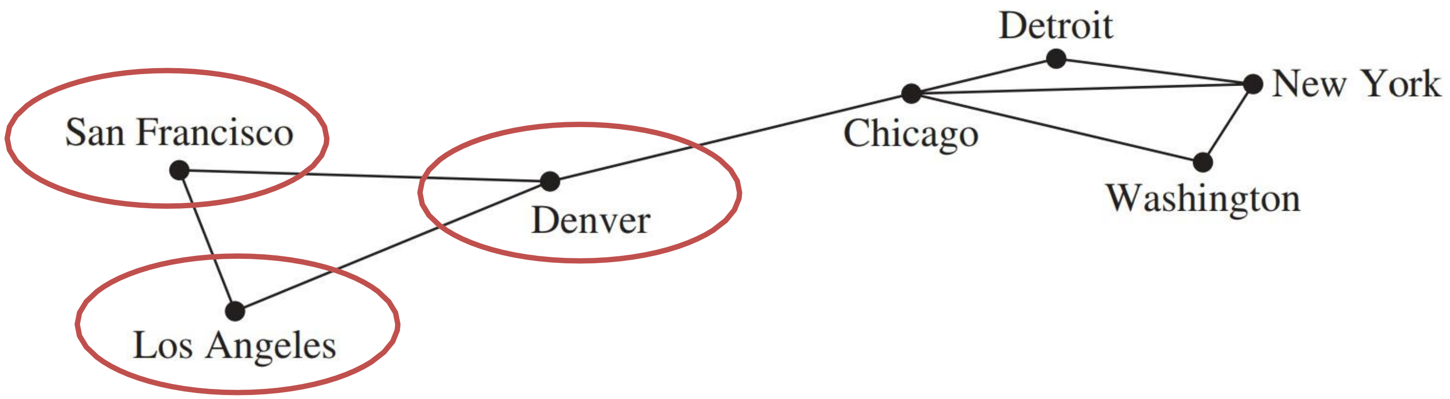


$K_{2,6}$

Some Special Simple Graphs (9/16)

New Graphs from Old:

Sometimes we need only part of a graph to solve a problem. For instance, we may care only about the part of a large computer network that involves the computer centers in some cities. Then we can ignore the other computer centers and all telephone lines not linking two of these specific computer centers.

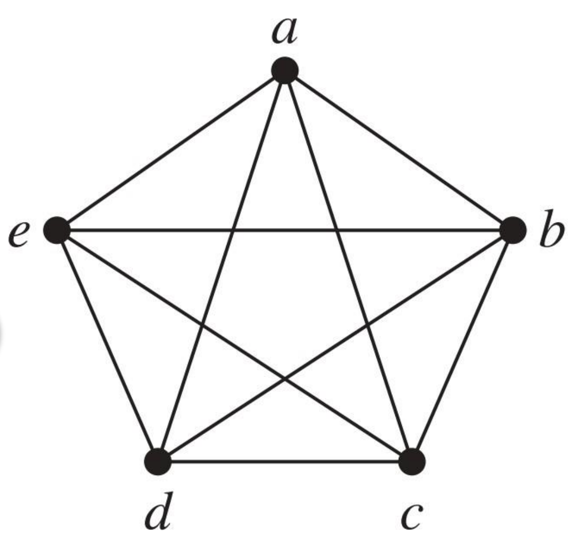


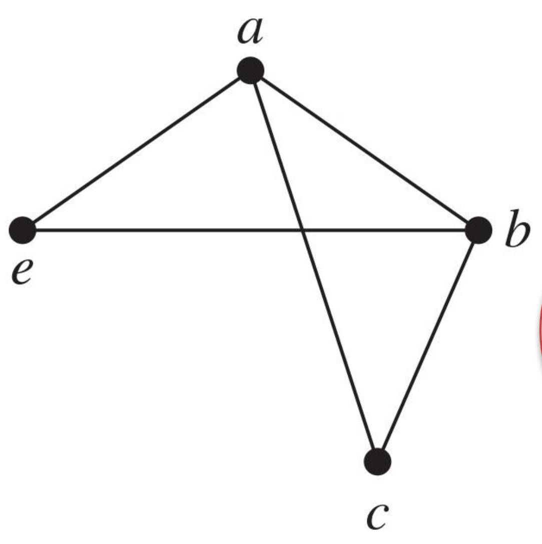
Some Special Simple Graphs (10/16)

Definition 7:

A ***subgraph*** of a graph $G = (V, E)$ is a graph $H = (W, F)$, where $W \subseteq V$ and $F \subseteq E$. A subgraph H of G is a ***proper subgraph*** of G if $H \neq G$.

Original Graph K_5



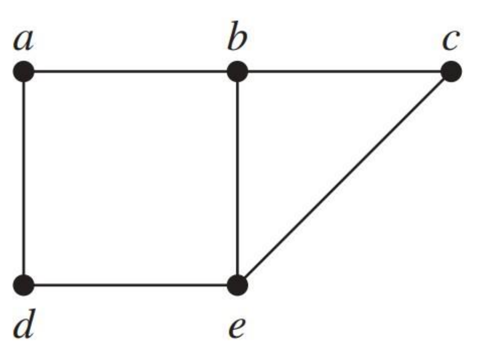


Proper Subgraph of The Original Graph

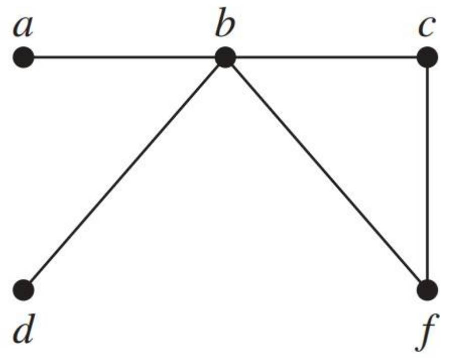
We can remove the vertices and edges from original graph

Some Special Simple Graphs (16/16)

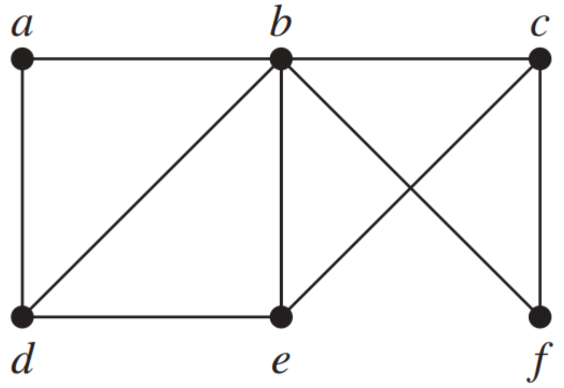
Graph Unions:



G_1



G_2



$G_1 \cup G_2$

Chapter 10: Graphs

- Graphs and Graph Models.
- Graph Terminology and Special Types of Graphs.
- Representing Graphs and Graph Isomorphism.
- Connectivity.

Representing Graphs (1/12)

Introduction:

There are many useful ways to represent graphs. In working with a graph, it is helpful to be able to choose its most appropriate representation. In this section, we will show how to represent graphs in several different ways.

1. Representing graph using ***adjacency list***.
2. Representing graph using ***adjacency matrix***.
3. Representing graph using ***incidence matrix***.

Representing Graphs (2/12)

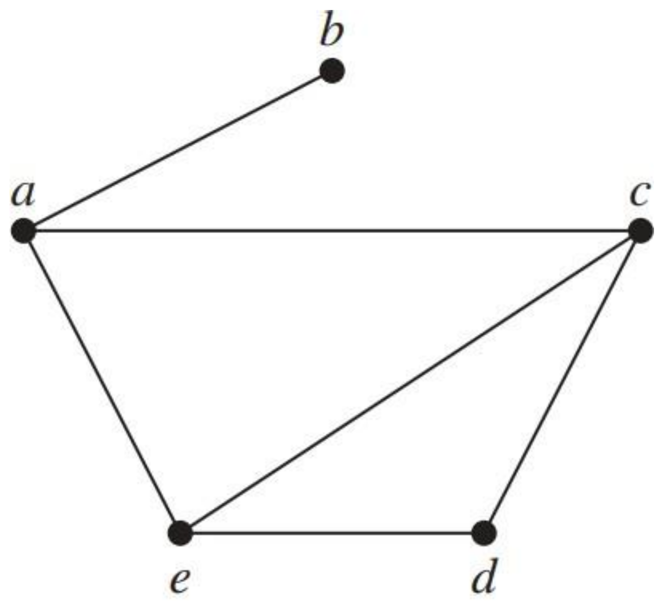
Adjacency Lists:

One way to represent a graph without multiple edges is to list all the *edges* of this graph. Another way to represent a graph with no multiple edges is to use ***adjacency lists***, which specify the vertices that are adjacent to each vertex of the graph.

Representing Graphs (3/12)

Adjacency Lists – Example 1:

Use adjacency lists to describe the following simple graph.



A simple graph.

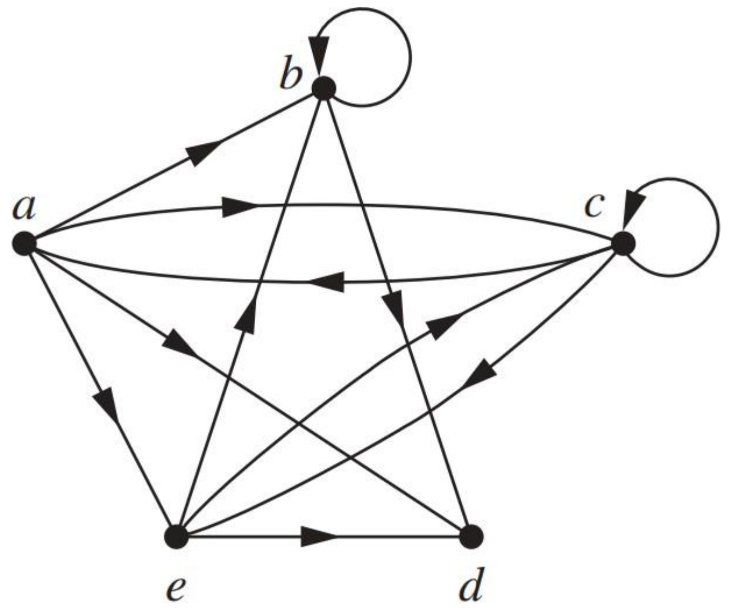
An Adjacency List for a Simple Graph.

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

Representing Graphs (4/12)

Adjacency Lists – Example 2:

Use adjacency lists to describe the following simple directed graph.



A directed graph.

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*

Representing Graphs (5/12)

Adjacency Matrices:

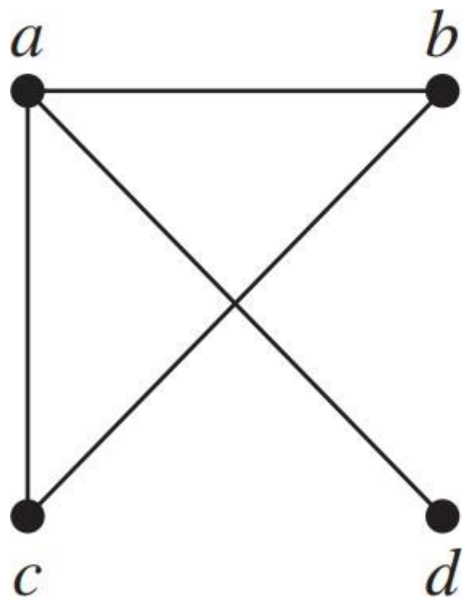
Suppose that $G = (V, E)$ is a simple graph where $|V| = n$. Suppose that the vertices of G are listed arbitrarily as $v_1, v_2, \ldots, v_n$. The adjacency matrix **A** (or $\mathbf{A}_G$) of G, with respect to this listing of the vertices, is the $n \times n$ zero–one matrix with 1 as its (i, j)th 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 its adjacency matrix is $\mathbf{A} = [a_{ij}]$, then

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

Representing Graphs (6/12)

Adjacency Matrices – Example 1:

Use an adjacency matrix to represent the following graph.

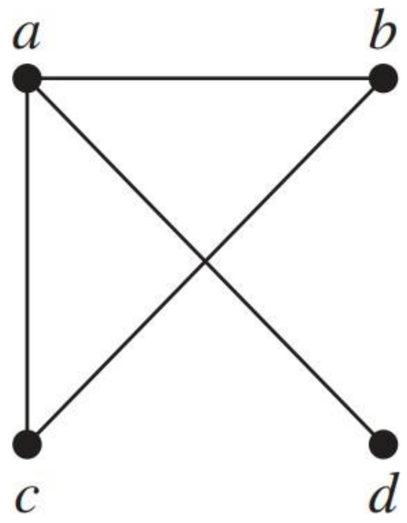


$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Representing Graphs (7/12)

Adjacency Matrices – Example 1:

Use an adjacency matrix to represent the following graph.



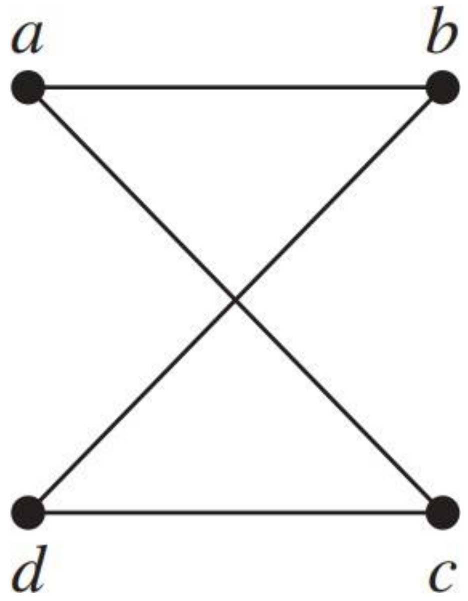
$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Note that an adjacency matrix of a graph is based on the ***ordering*** chosen for the ***vertices***. Hence, there may be as many as $n!$ different adjacency matrices for a graph with n vertices, because there are $n!$ different orderings of n vertices.

Representing Graphs (8/12)

Adjacency Matrices – Example 2:

Use an adjacency matrix to represent the following graph.

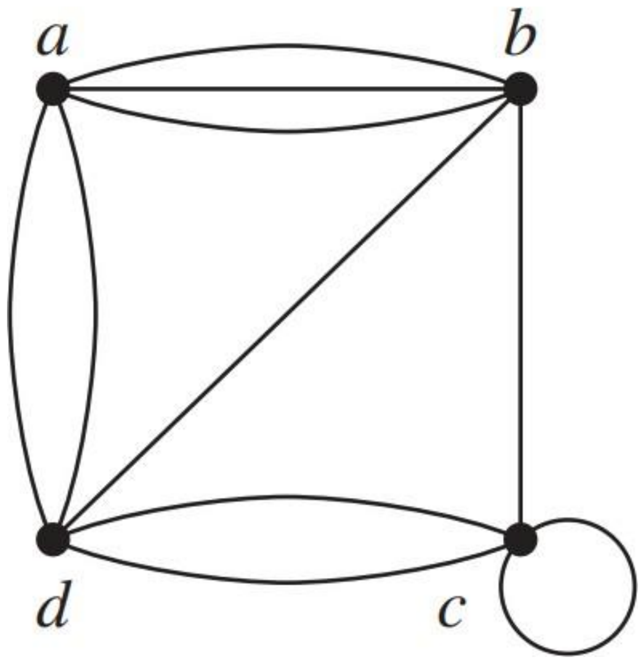


$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Representing Graphs (9/12)

Adjacency Matrices – Example 3:

Use an adjacency matrix to represent the following pseudograph.



$$\begin{bmatrix} 0 & 3 & 0 & 2 \\ 3 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 \\ 2 & 1 & 2 & 0 \end{bmatrix}$$

Representing Graphs (10/12)

Incidence Matrices:

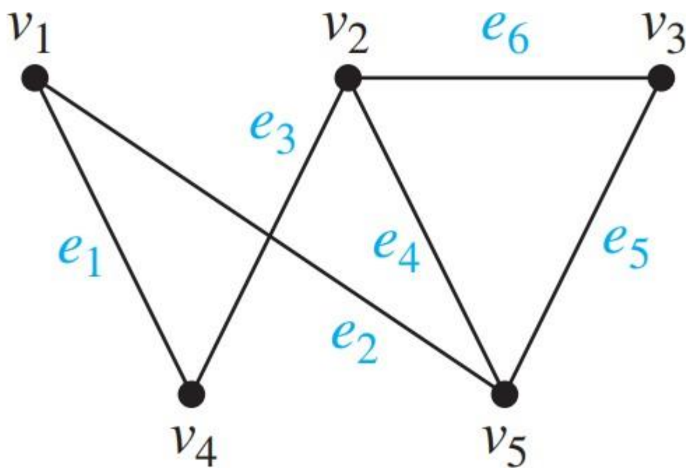
Let $G = (V, E)$ be an undirected graph. Suppose that $v_1, v_2, \dots, v_n$ are the vertices and $e_1, e_2, \dots, e_m$ are the edges of G. Then the incidence matrix with respect to this ordering of V and E is the $n \times m$ matrix $\mathbf{M} = [a_{ij}]$, where

$$a_{ij} = \begin{cases} 1 & \text{when edge } e_j \text{ is incident with } v_i, \\ 0 & \text{otherwise.} \end{cases}$$

Representing Graphs (11/12)

Incidence Matrices – Example 1:

Represent the following graph with an incidence matrix.

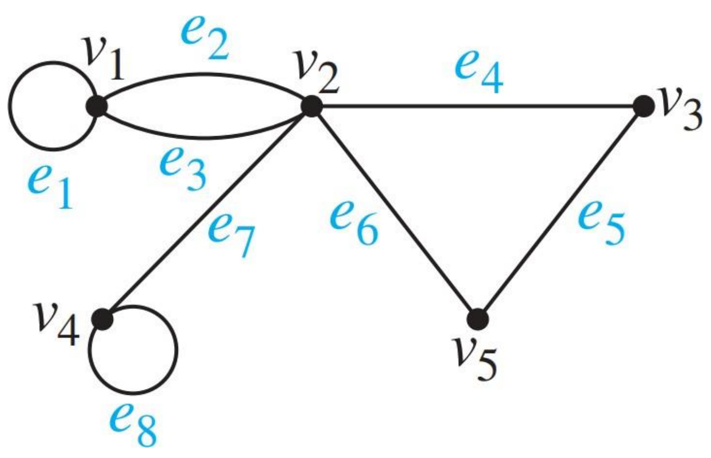


$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

Representing Graphs (12/12)

Incidence Matrices – Example 2:

Represent the pseudograph with an incidence matrix.



$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{cccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \end{array} \\ \left[\begin{array}{cccccccc} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{array}\right] \end{array}$$

Isomorphism of Graphs (1/20)

Introduction:

- The word isomorphism comes from the Greek roots *isos* for “equal” and *morphe* for “form.”
- Graphs with the *same structure*.
- When two simple graphs are ***isomorphic***, there is a one-to-one correspondence between *vertices* of the two graphs that preserves the adjacency relationship.

Isomorphism of Graphs (2/20)

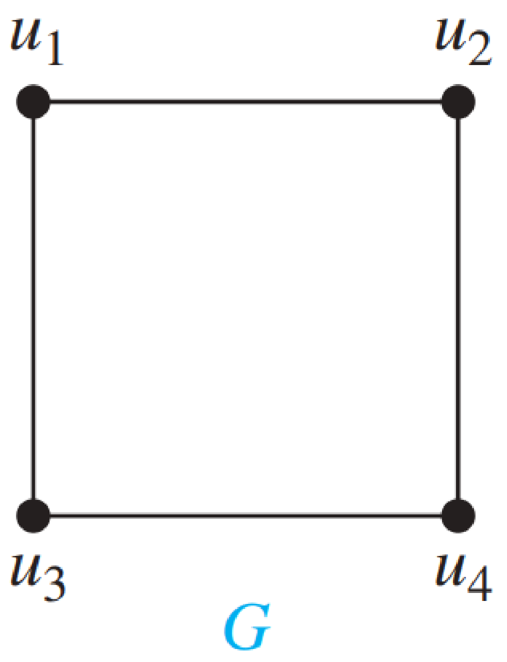
Definition:

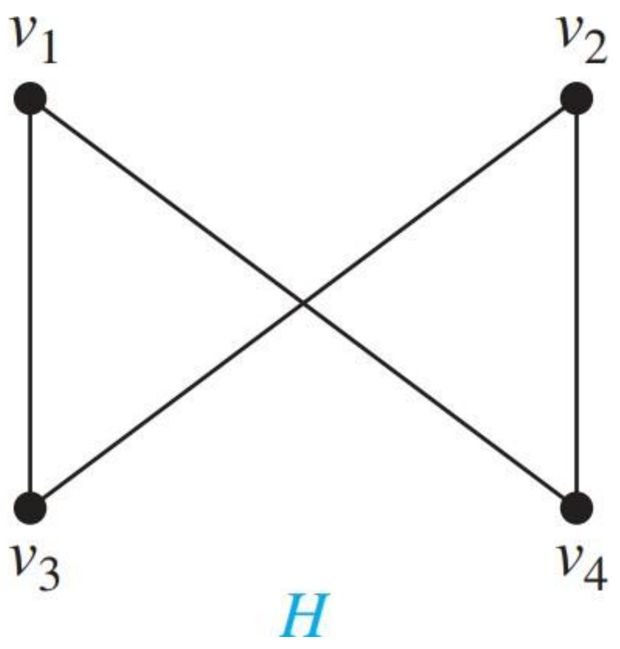
The simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are isomorphic if there exists a one-to-one and onto function f from V_1 to V_2 with the property that a and b are adjacent in G_1 if and only if $f(a)$ and $f(b)$ are adjacent in G_2, for all a and b in V_1. Such a function f is called an ***isomorphism***. Two simple graphs that are **not** isomorphic are called ***nonisomorphic***.

Isomorphism of Graphs (3/20)

Example 1:

Show that the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic.

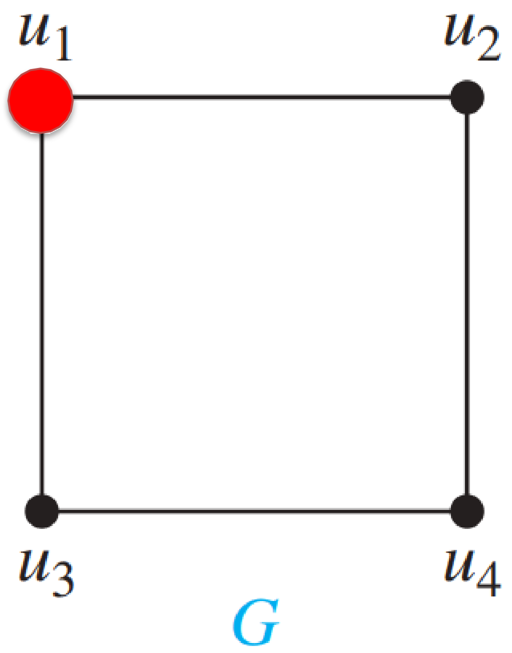


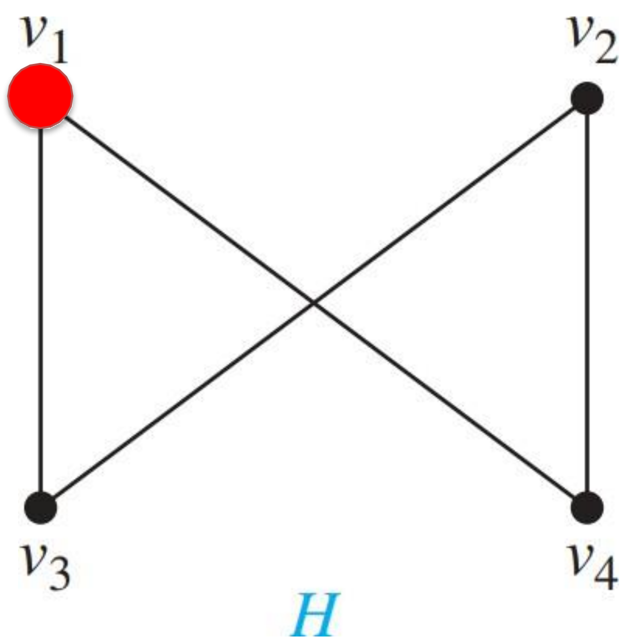


Isomorphism of Graphs (3/20)

Example 1:

Show that the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic.

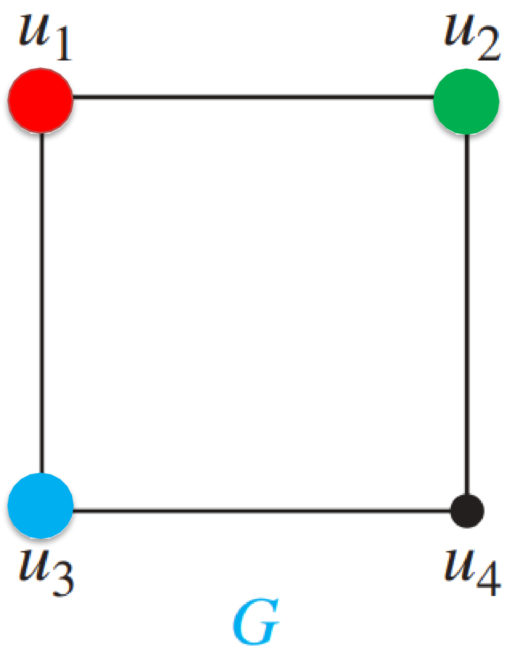


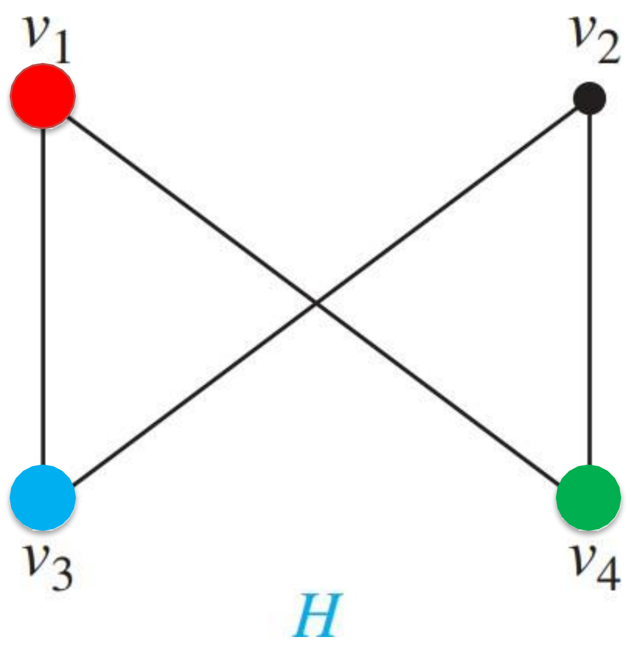


Isomorphism of Graphs (3/20)

Example 1:

Show that the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic.

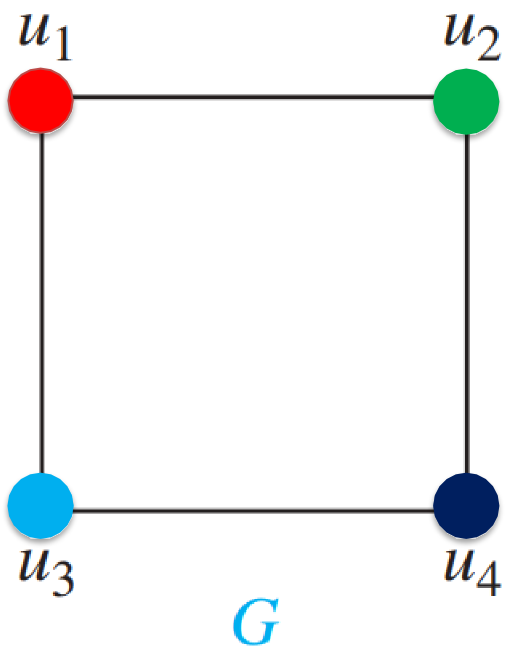


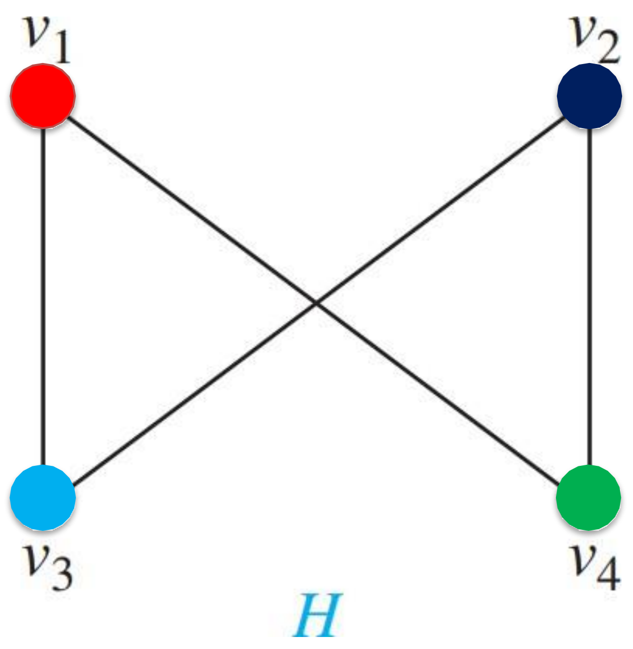


Isomorphism of Graphs (3/20)

Example 1:

Show that the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic.

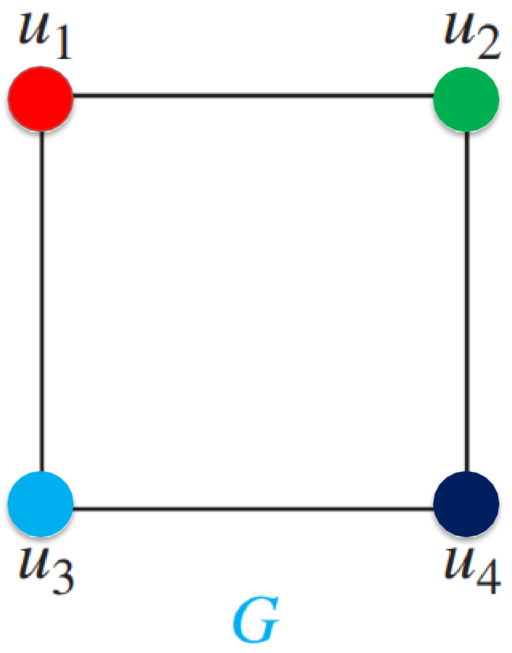


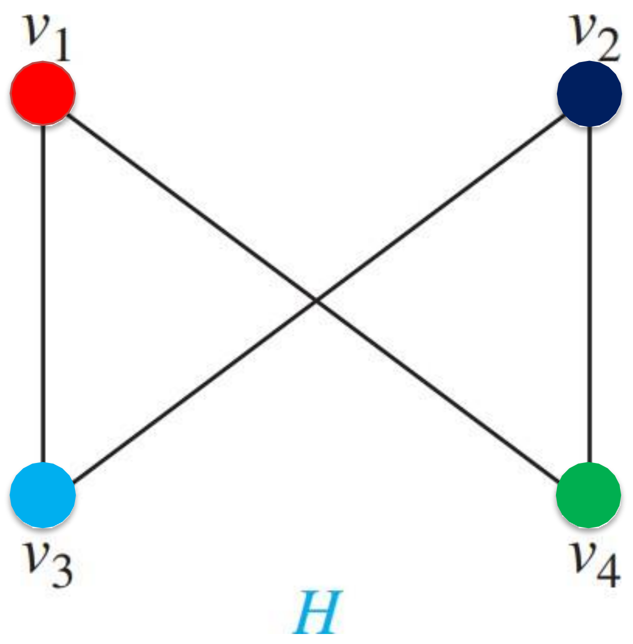


Isomorphism of Graphs (4/20)

Example 1:

Show that the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic.



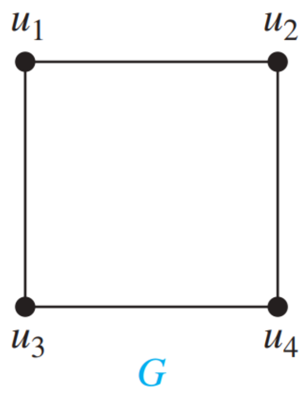


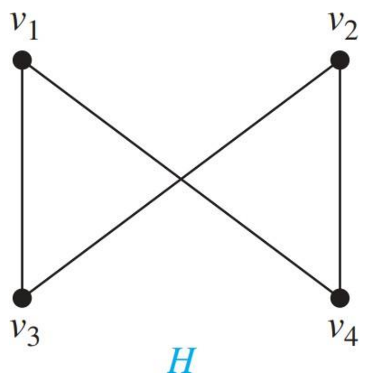
The function f with $f(u_1) = v_1$, $f(u_2) = v_4$, $f(u_3) = v_3$, and $f(u_4) = v_2$ is a *one-to-one correspondence* between V and W.

Isomorphism of Graphs (5/20)

Example 1:

Show that the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic.





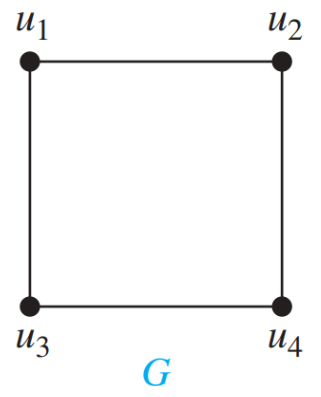
$f(u_1) = v_1$, $f(u_2) = v_4$, $f(u_3) = v_3$, and $f(u_4) = v_2$

The adjacent vertices in G are $\boxed{\boldsymbol{u_1} \textbf{ and } \boldsymbol{u_2},}$ $\boldsymbol{u_1}$ **and** $\boldsymbol{u_3}$, $\boldsymbol{u_2}$ **and** $\boldsymbol{u_4}$, and $\boldsymbol{u_3}$ **and** $\boldsymbol{u_4}$, and each of the pairs $\boxed{\boldsymbol{f(u_1) = v_1} \textbf{ and } \boldsymbol{f(u_2) = v_4},}$ $\boldsymbol{f(u_1) = v_1}$ **and** $\boldsymbol{f(u_3) = v_3}$, $\boldsymbol{f(u_2) = v_4}$ **and** $\boldsymbol{f(u_4) = v_2}$, and $\boldsymbol{f(u_3) = v_3}$ **and** $\boldsymbol{f(u_4) = v_2}$ consists of two adjacent vertices in H.

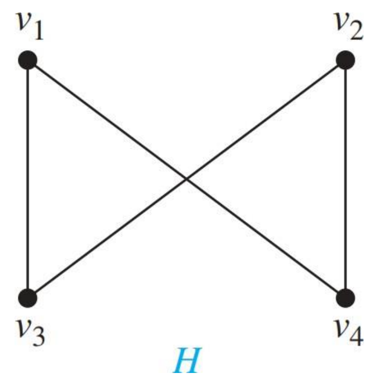
Isomorphism of Graphs (6/20)

Example 1:

Show that the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic.



Test using adjacency matrices



$\boldsymbol{f(u_1) = v_1,\ f(u_2) = v_4,\ f(u_3) = v_3,}$ **and** $\boldsymbol{f(u_4) = v_2}$

$$\begin{array}{c|cccc} G & u_1 & u_2 & u_3 & u_4 \\ \hline u_1 & 0 & 1 & 1 & 0 \\ u_2 & 1 & 0 & 0 & 1 \\ u_3 & 1 & 0 & 0 & 1 \\ u_4 & 0 & 1 & 1 & 0 \end{array} = \begin{array}{c|cccc} H & v_1 & v_4 & v_3 & v_2 \\ \hline v_1 & 0 & 1 & 1 & 0 \\ v_4 & 1 & 0 & 0 & 1 \\ v_3 & 1 & 0 & 0 & 1 \\ v_2 & 0 & 1 & 1 & 0 \end{array}$$

Isomorphism of Graphs (7/20)

Isomorphic or Not (1/2):

It is often difficult to determine whether two simple graphs are isomorphic. There are $n!$ possible one-to-one correspondences between the vertex sets of two simple

- graphs with n vertices. Testing each such correspondence to see whether it preserves adjacency and nonadjacency is impractical if n is at all large.

Isomorphism of Graphs (7/20)

Isomorphic or Not (2/2):

Sometimes it is not hard to show that two graphs are not isomorphic. In particular, we can show that two graphs are not isomorphic if we can find a property only one of the two graphs has.

Isomorphism of Graphs (8/20)

Graph Invariant:

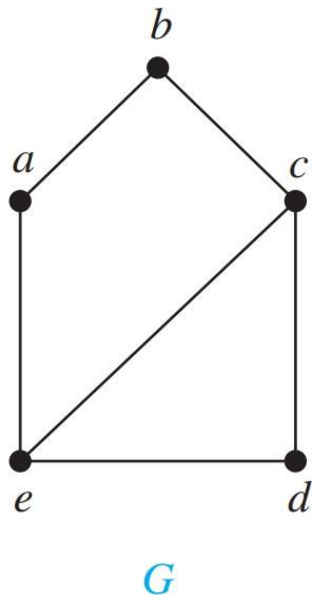
A property preserved by isomorphism of graphs is called a ***graph invariant***.

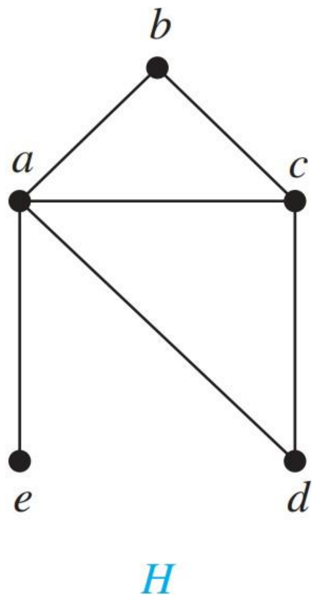
1. Same number of vertices,
2. Same number of edges,
3. Same degrees of the vertices,
4. Adjacency test, …

Isomorphism of Graphs (9/20)

Example 2:

Determine whether the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic or not.

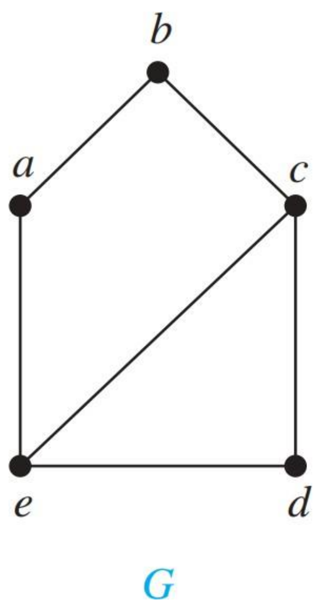


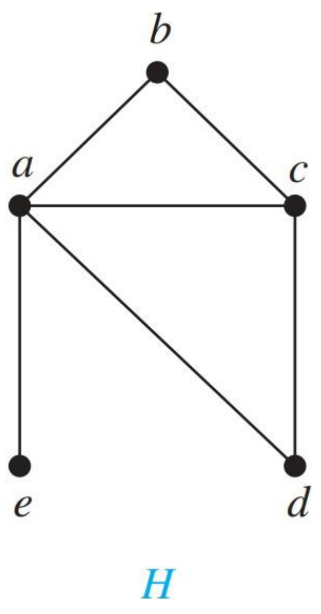


Isomorphism of Graphs (10/20)

Example 2:

Determine whether the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic or not.



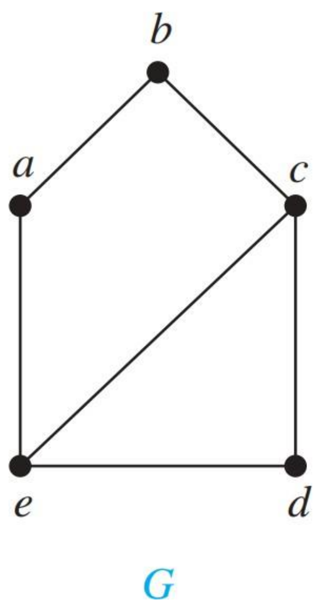


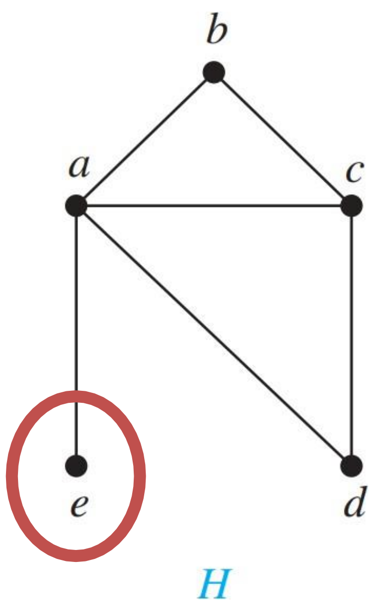
1. Same number of vertices (---)
2. Same number of edges (---)
3. Same degrees of the vertices (---)

Isomorphism of Graphs (11/20)

Example 2:

Determine whether the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic or not.





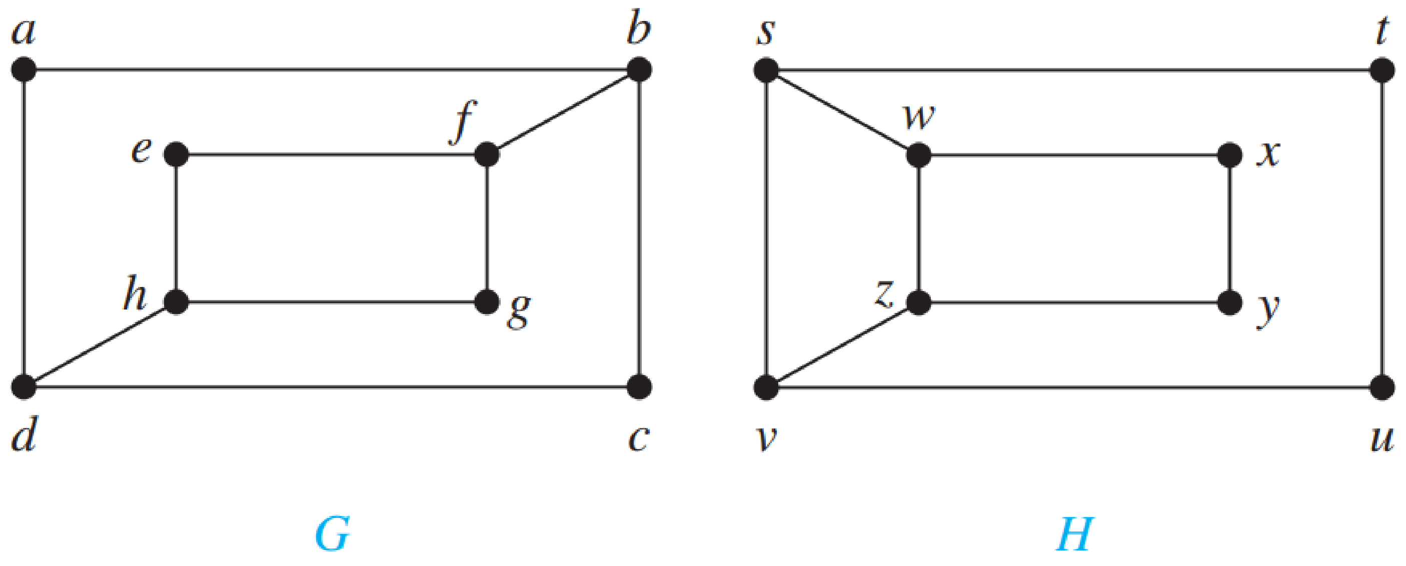
1. Same number of vertices (Yes)
2. Same number of edges (Yes)
3. Same degrees of the vertices (No)

   For instance, H has a vertex of degree one, namely, e, whereas G has no vertices of degree one. It follows that G and H are ***not isomorphic***.

Isomorphism of Graphs (12/20)

Example 3:

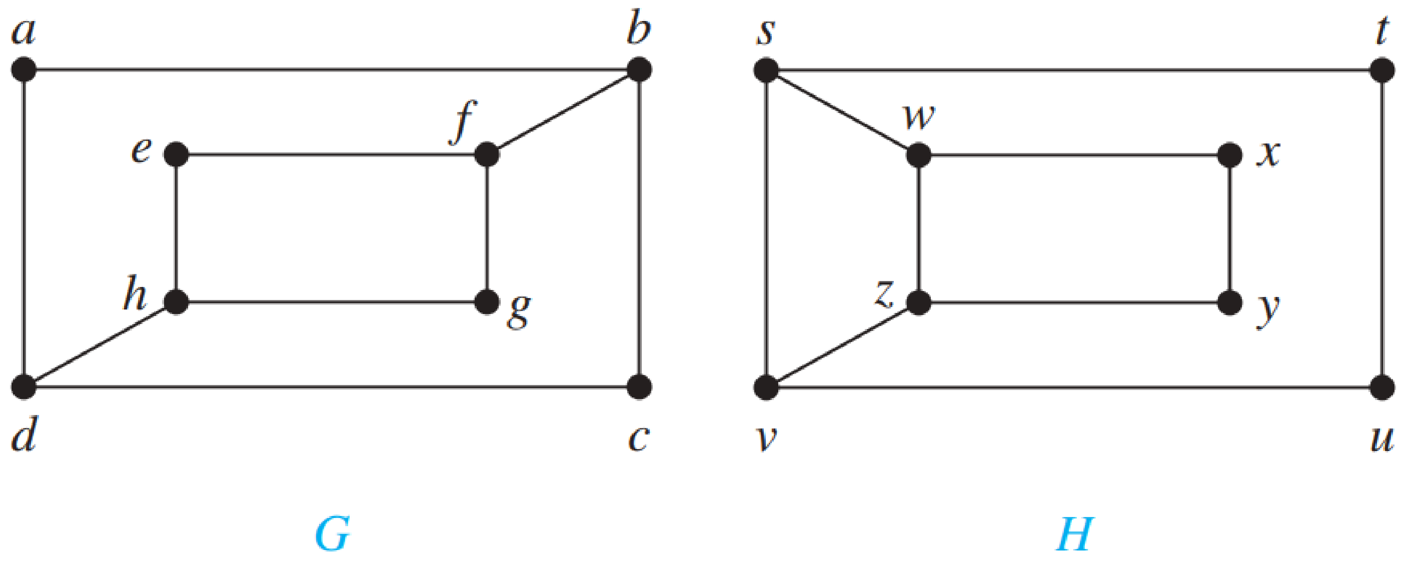
Determine whether the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic or not.



Isomorphism of Graphs (13/20)

Example 3:

Determine whether the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic or not.

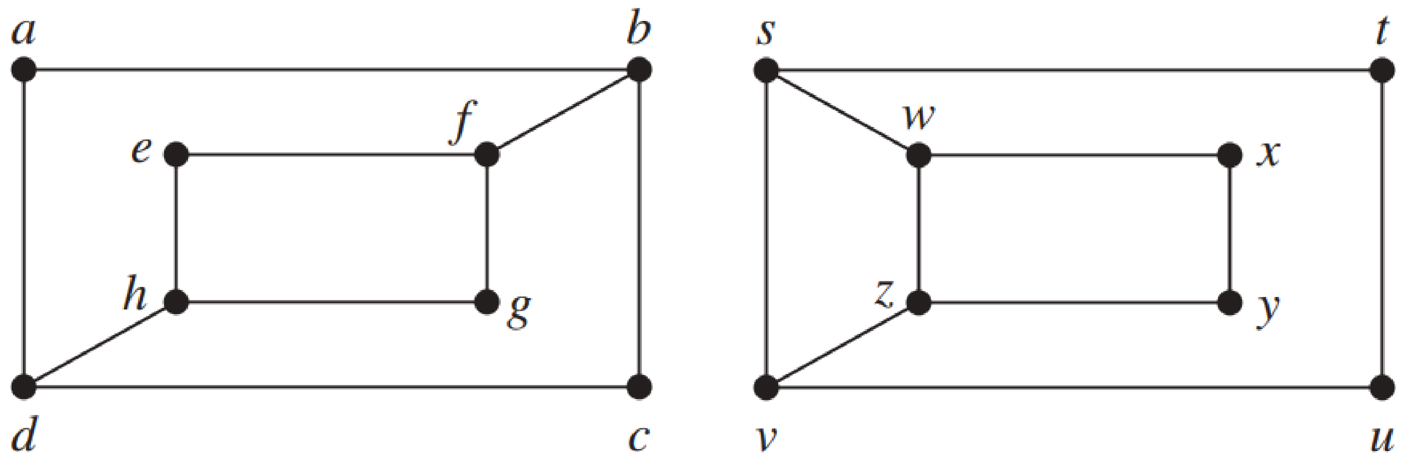


1. Same number of vertices (Yes)

Isomorphism of Graphs (14/20)

Example 3:

Determine whether the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic or not.



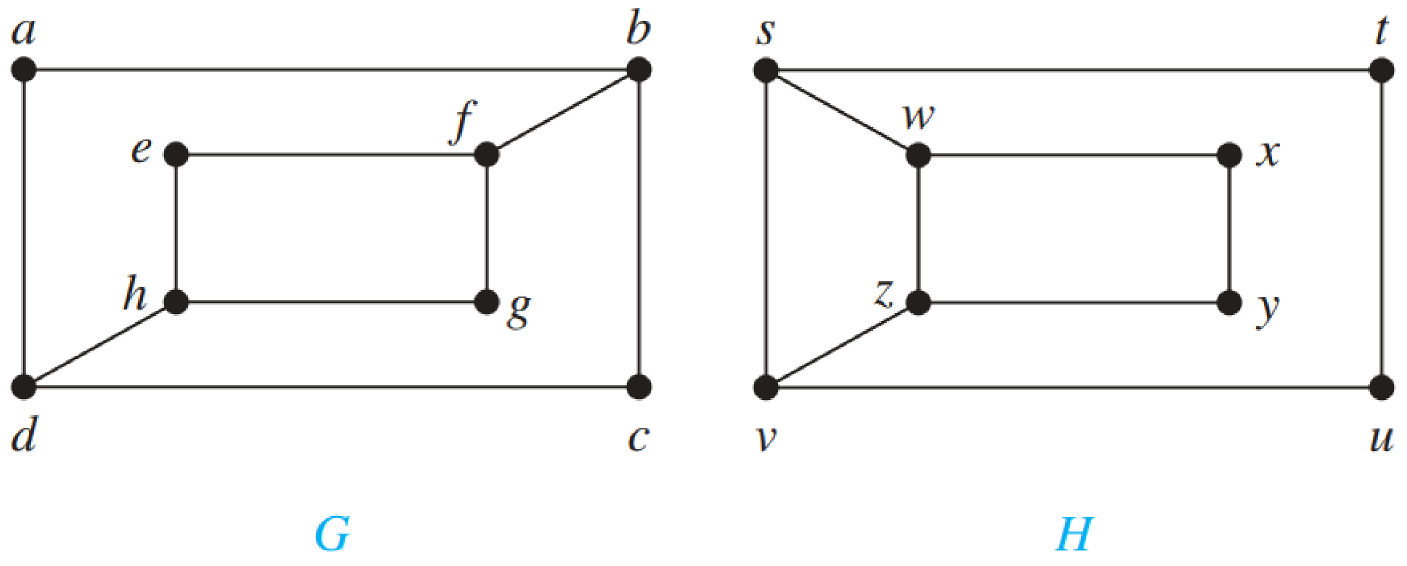
G *H*

2. Same number of edges (Yes)

Isomorphism of Graphs (15/20)

Example 3:

Determine whether the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic or not.

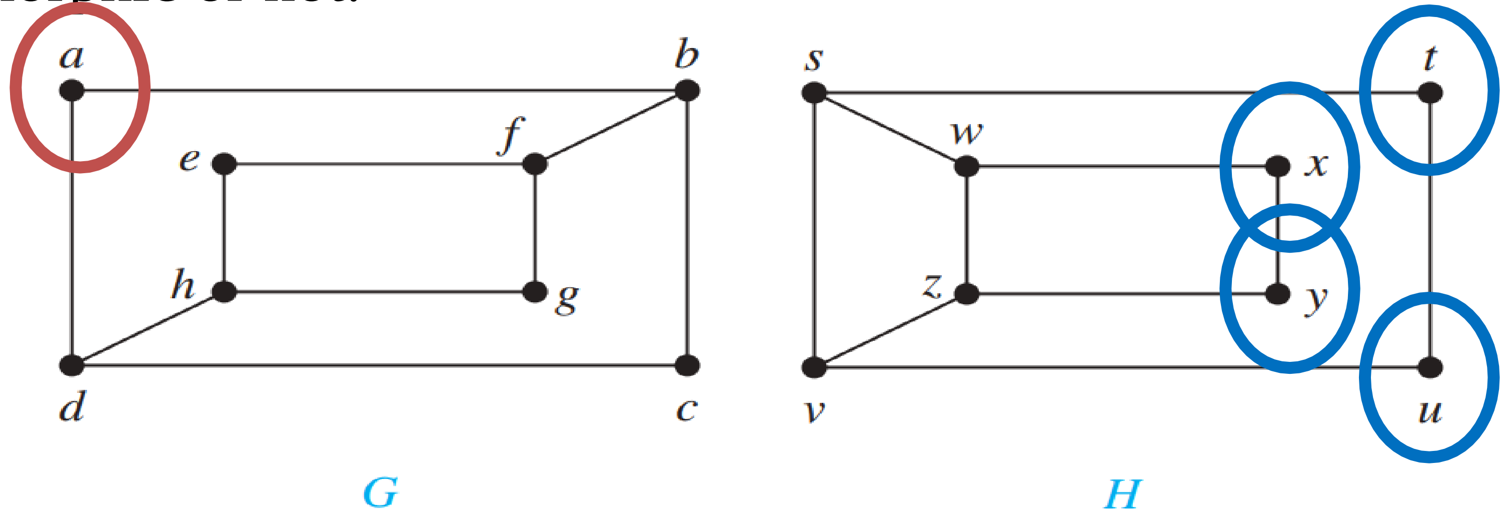


3. Same degrees of the vertices (Yes)

Isomorphism of Graphs (16/20)

Example 3:

Determine whether the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic or not.

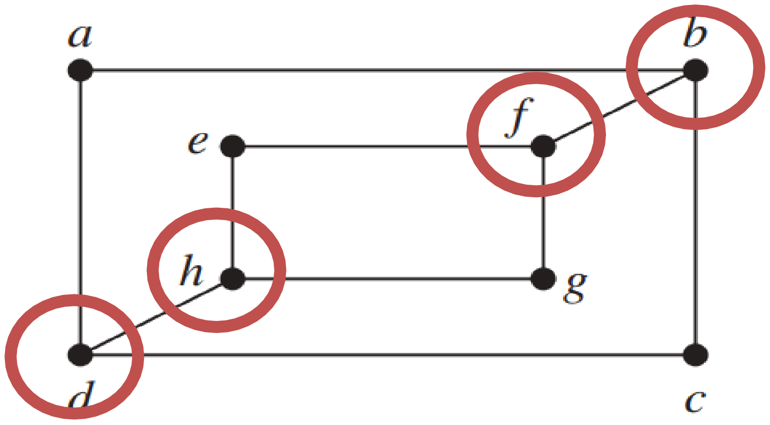


However, G and H are ***not isomorphic***. To see this, note that because $\deg(a) = 2$ in G, a must correspond to either $\boldsymbol{t}$, $\boldsymbol{u}$, $\boldsymbol{x}$, or $\boldsymbol{y}$ in H, because these are the vertices of degree two in H. However, each of these four vertices in H is adjacent to another vertex of degree two in H, which is not true for a in G.

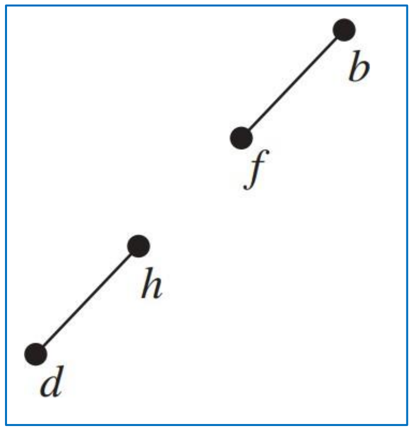
Isomorphism of Graphs (17/20)

Example 3:

Determine whether the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic or not.



G

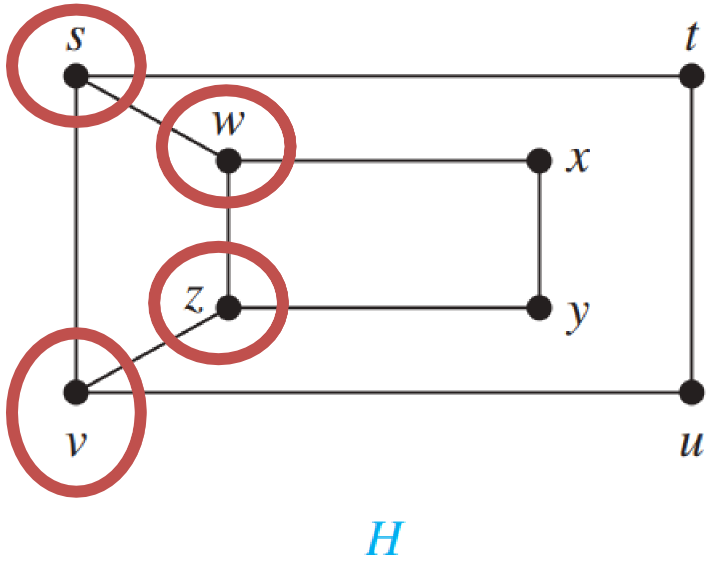


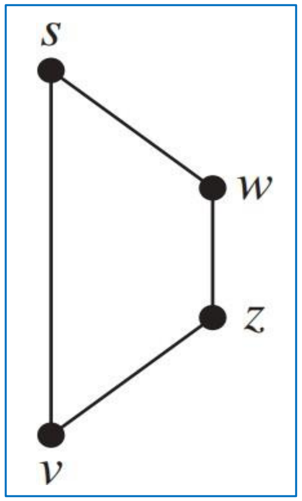
The ***subgraph*** of G made up of vertices of **degree three** and the edges connecting them

Isomorphism of Graphs (18/20)

Example 3:

Determine whether the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic or not.



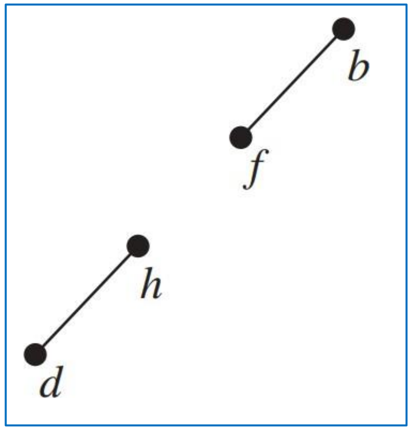


The ***subgraph*** of H made up of vertices of **degree three** and the edges connecting them

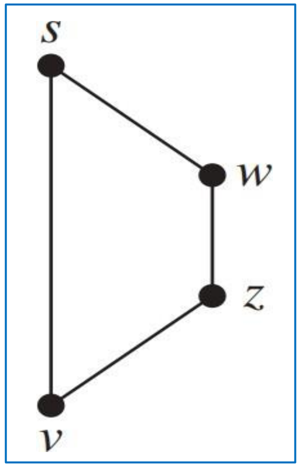
Isomorphism of Graphs (19/20)

Example 3:

Determine whether the graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic or not.



1. Same number of vertices (Yes)
2. Same number of edges (No)
   - It follows that the two subgraphs are ***not isomorphic***.



Another way to see that G and H are not isomorphic is to note that the ***subgraphs*** of G and H made up of vertices of **degree three** and the edges connecting them must be isomorphic if these two graphs are isomorphic. However, these subgraphs are not isomorphic.

Isomorphism of Graphs (20/20)

Applications of Graph Isomorphisms:

Graph isomorphisms used in chemistry and to the design of electronic circuits, and other areas including bioinformatics and computer vision.

NAUTY (no **aut**omorphisms, **y**es?) software can be used to determine whether two graphs are isomorphic or not.

[Ref] McKay, B.D. and Piperno, A., Practical Graph Isomorphism, II, Journal of Symbolic Computation, 60 (2014), pp. 94-112. https://pallini.di.uniroma1.it/index.html

Connectivity (1/20)

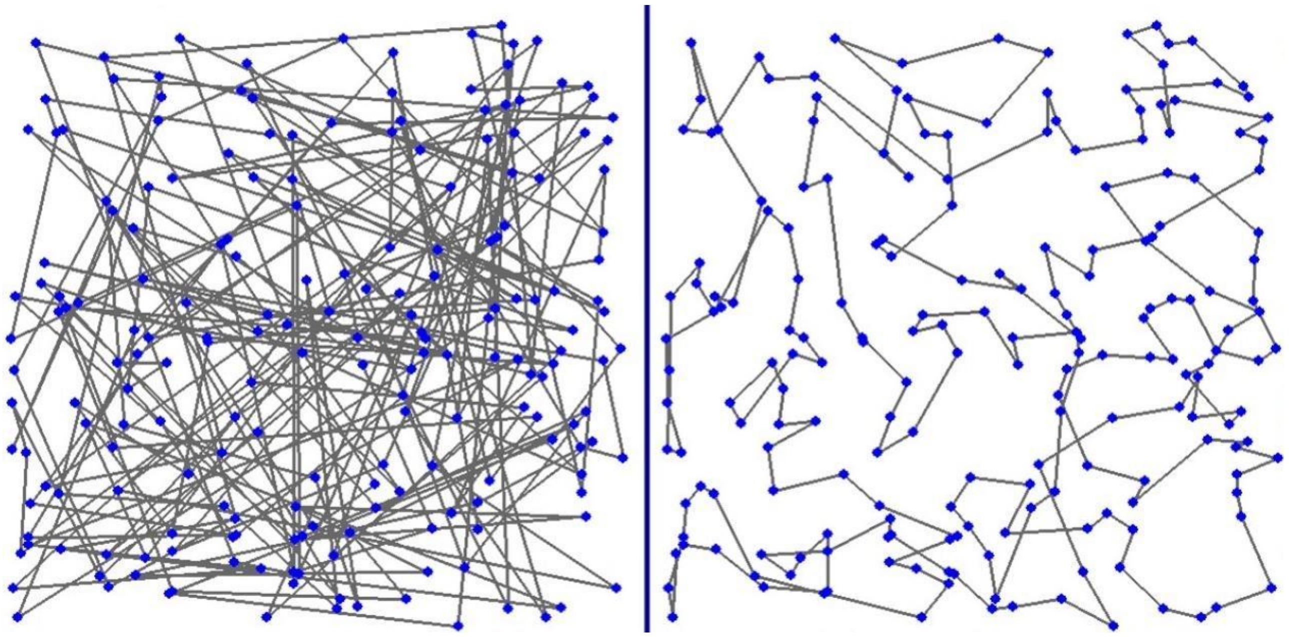
Introduction:

Many problems can be modeled with paths formed by traveling along the edges of graphs.

- Shortest-Path Problems
  - ➢ *Traveling salesman problem* (TSP) (also called the *traveling salesperson problem*), which asks for an order in which a salesperson should visit each of the cities on his route exactly once so that he travels the minimum total distance.

Connectivity (2/20)

TSP:



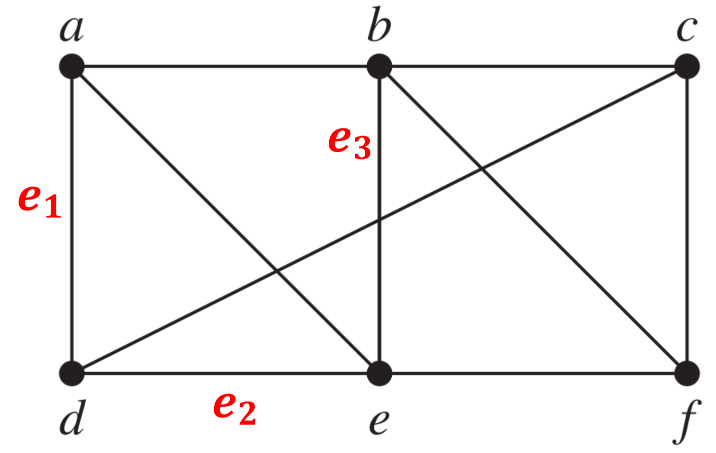
A **path** is a sequence of edges that begins at a vertex of a graph and travels from vertex to vertex along edges of the graph.

Connectivity (3/20)

Definition 1 (1/2):

Let n be a nonnegative integer and G an undirected graph. A ***path*** of length $\boldsymbol{n}$ from $\boldsymbol{u}$ to $\boldsymbol{v}$ in G is a sequence of n edges $e_1, \dots, e_n$ of G for which there exists a sequence $x_0 = u$, $x_1, \dots, x_{n-1}$, $x_n = v$ of vertices such that e_i has, for $i = 1, \dots, n$, the endpoints x_{i-1} and x_i. When the graph is simple, we denote this path by its vertex sequence $x_0, x_1, \dots, x_n$.

A ***path*** of length **3** from $\boldsymbol{a}$ to $\boldsymbol{b}$:
- Sequence of 3 edges e_1, e_2, e_3.
- Sequence a, d, e, b of vertices.



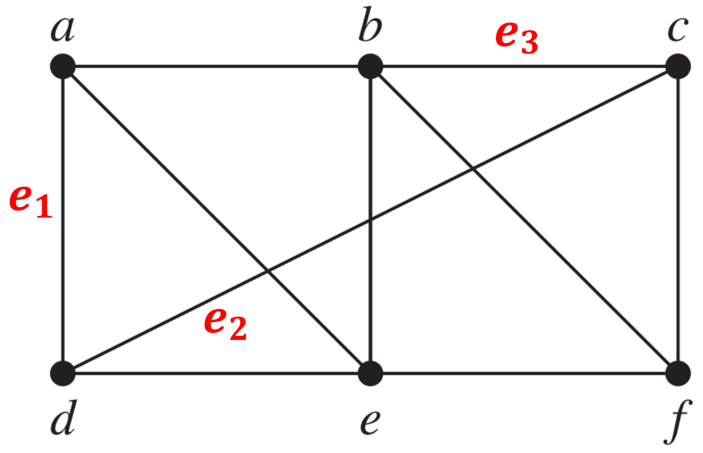
Connectivity (3/20)

Definition 1 (1/2):

Let n be a nonnegative integer and G an undirected graph. A ***path*** of length $\boldsymbol{n}$ from $\boldsymbol{u}$ to $\boldsymbol{v}$ in G is a sequence of n edges $e_1, \dots, e_n$ of G for which there exists a sequence $x_0 = u$, $x_1, \dots, x_{n-1}$, $x_n = v$ of vertices such that e_i has, for $i = 1, \dots, n$, the endpoints x_{i-1} and x_i. When the graph is simple, we denote this path by its vertex sequence $x_0, x_1, \dots, x_n$.

A ***path*** of length **3** from $\boldsymbol{a}$ to $\boldsymbol{b}$:

- Sequence of 3 edges e_1, e_2, e_3.
- Sequence a, d, c, b of vertices.

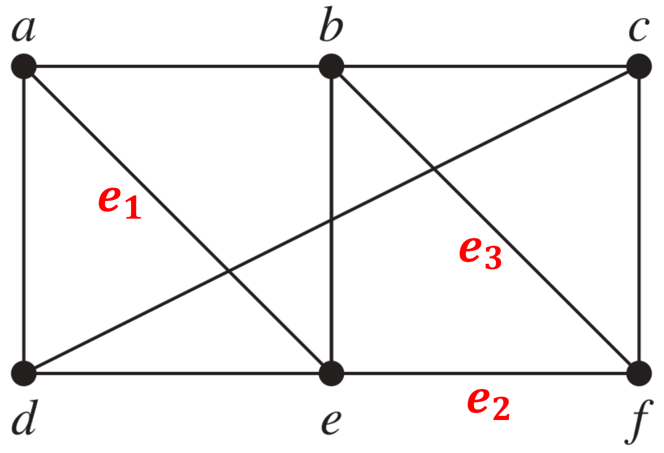


Connectivity (3/20)

Definition 1 (1/2):

Let n be a nonnegative integer and G an undirected graph. A ***path*** of length $\boldsymbol{n}$ from $\boldsymbol{u}$ to $\boldsymbol{v}$ in G is a sequence of n edges $e_1, \ldots, e_n$ of G for which there exists a sequence $x_0 = u$, $x_1, \ldots, x_{n-1}$, $x_n = v$ of vertices such that e_i has, for $i = 1, \ldots, n$, the endpoints x_{i-1} and x_i. When the graph is simple, we denote this path by its vertex sequence $x_0, x_1, \ldots, x_n$.

A ***path*** of length **3** from $\boldsymbol{a}$ to $\boldsymbol{b}$:
- Sequence of 3 edges e_1, e_2, e_3.
- Sequence a, e, f, b of vertices.



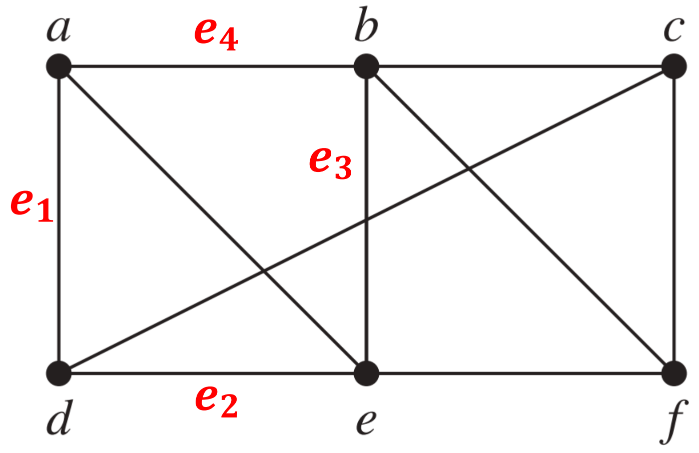
Connectivity (3/20)

Definition 1 (2/2):

The path is a ***circuit*** if it begins and ends at the same vertex, that is, if $u = v$, and has length greater than zero. The path or circuit is said to *pass through* the vertices $x_1, x_2, \ldots, x_{n-1}$ or *traverse* the edges $e_1, e_2, \ldots, e_n$. A path or circuit is *simple* if it does not contain the same edge more than once.

A ***circuit*** of length **4** from ***a*** to ***a***:

- Sequence of 4 edges e_1, e_2, e_3, e_4.
- Sequence a, d, e, b, a of vertices.



Connectivity (3/20)

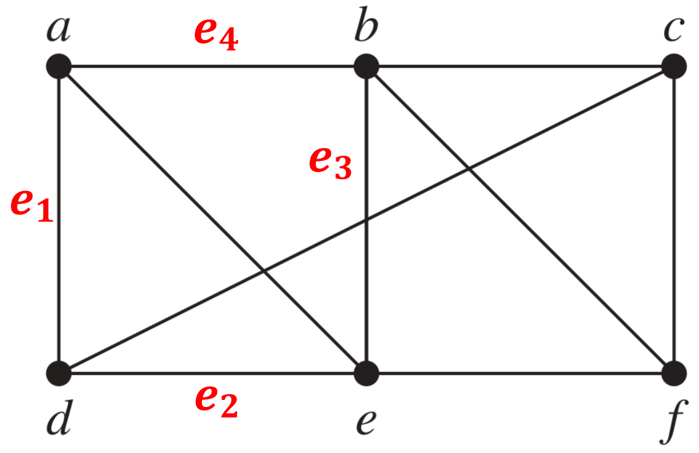
Definition 1 (2/2):

The path is a ***circuit*** if it begins and ends at the same vertex, that is, if $u = v$, and has length greater than zero. The path or circuit is said to *pass through* the vertices $x_1, x_2, \ldots, x_{n-1}$ or *traverse* the edges $e_1, e_2, \ldots, e_n$. A path or circuit is *simple* if it does not contain the same edge more than once.

Simple path with circuit

A ***circuit*** of length **4** from $\boldsymbol{a}$ to $\boldsymbol{a}$:

- Sequence of 4 edges e_1, e_2, e_3, e_4.
- Sequence a, d, e, b, a of vertices.



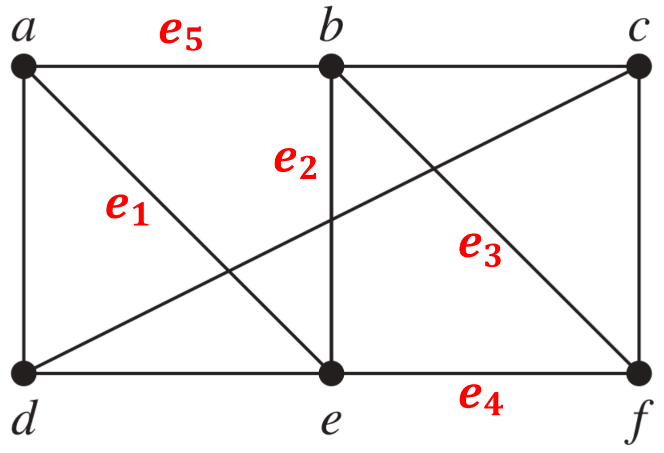
Connectivity (3/20)

Definition 1 (2/2):

The path is a ***circuit*** if it begins and ends at the same vertex, that is, if $u = v$, and has length greater than zero. The path or circuit is said to *pass through* the vertices $x_1, x_2, \ldots, x_{n-1}$ or *traverse* the edges $e_1, e_2, \ldots, e_n$. A path or circuit is *simple* if it does not contain the same edge more than once.

A ***circuit*** of length **6** from $\boldsymbol{a}$ to $\boldsymbol{a}$:

- Sequence of 6 edges: $e_1, e_2, e_3, e_4, e_2, e_5$.
- Sequence a, e, b, f, e, b, a of vertices.



Connectivity (3/20)

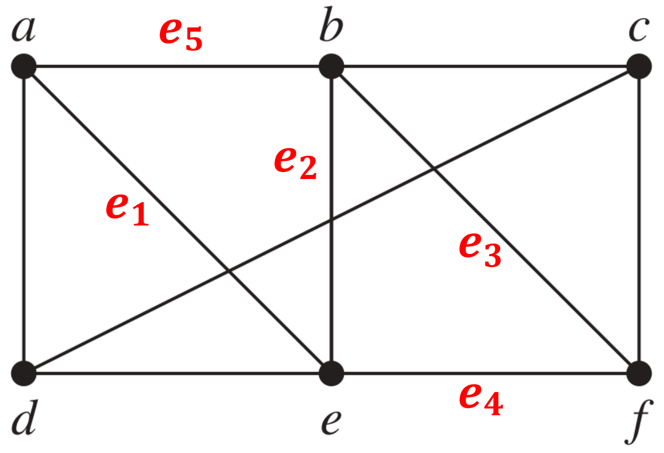
Definition 1 (2/2):

The path is a ***circuit*** if it begins and ends at the same vertex, that is, if $u = v$, and has length greater than zero. The path or circuit is said to *pass through* the vertices $x_1, x_2, \dots, x_{n-1}$ or *traverse* the edges $e_1, e_2, \dots, e_n$. A path or circuit is *simple* if it does not contain the same edge more than once.

Not simple path with circuit

A ***circuit*** of length **6** from ***a*** to ***a***:

- Sequence of 6 edges: $e_1, \boldsymbol{e_2}, e_3, e_4, \boldsymbol{e_2}, e_5$.
- Sequence a, e, b, f, e, b, a of vertices.



Connectivity (3/20)

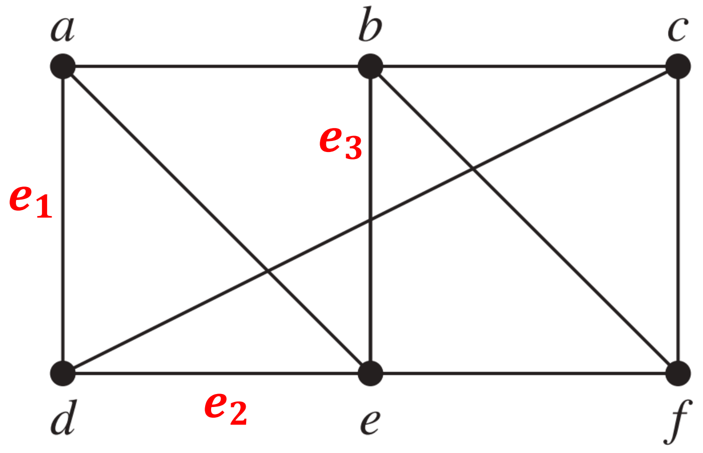
Definition 1 (2/2):

The path is a ***circuit*** if it begins and ends at the same vertex, that is, if $u = v$, and has length greater than zero. The path or circuit is said to *pass through* the vertices $x_1, x_2, \ldots, x_{n-1}$ or *traverse* the edges $e_1, e_2, \ldots, e_n$. A path or circuit is *simple* if it does not contain the same edge more than once.

Simple path

A ***path*** of length **3** from $\boldsymbol{a}$ to $\boldsymbol{b}$:

- Sequence of 3 edges e_1, e_2, e_3.
- Sequence a, d, e, b of vertices.



Connectivity (3/20)

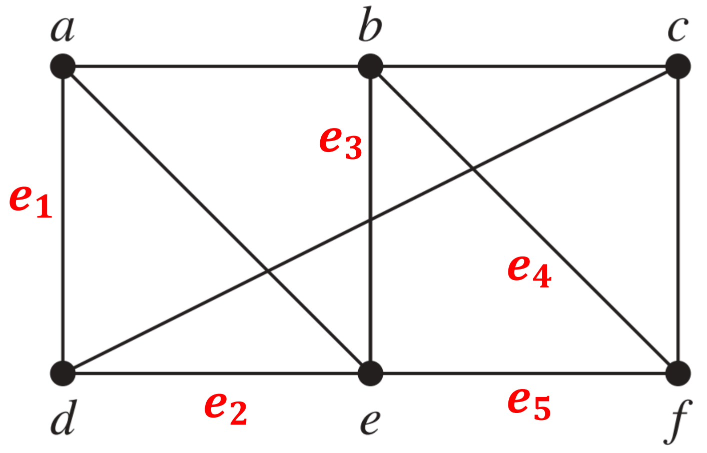
Definition 1 (2/2):

The path is a ***circuit*** if it begins and ends at the same vertex, that is, if $u = v$, and has length greater than zero. The path or circuit is said to *pass through* the vertices $x_1, x_2, \ldots, x_{n-1}$ or *traverse* the edges $e_1, e_2, \ldots, e_n$. A path or circuit is *simple* if it does not contain the same edge more than once.

Not simple path

A ***path*** of length **6** from $\boldsymbol{a}$ to $\boldsymbol{d}$:

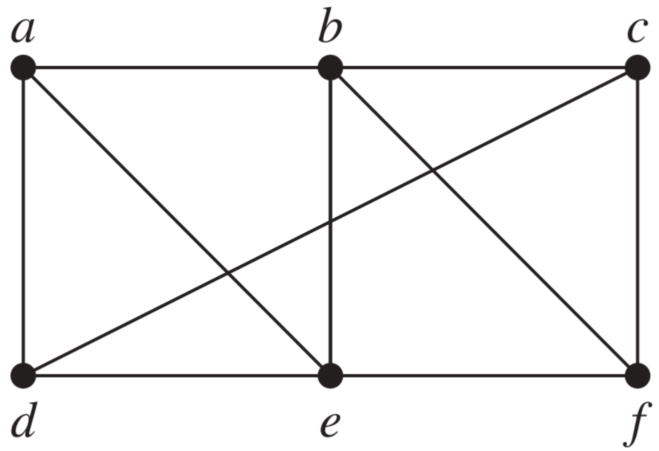
- Sequence of 6 edges: $e_1, \boldsymbol{e_2}, e_3, e_4, e_5, \boldsymbol{e_2}$.
- Sequence a, d, e, b, f, e, d of vertices.



Connectivity (4/20)

Example 1:

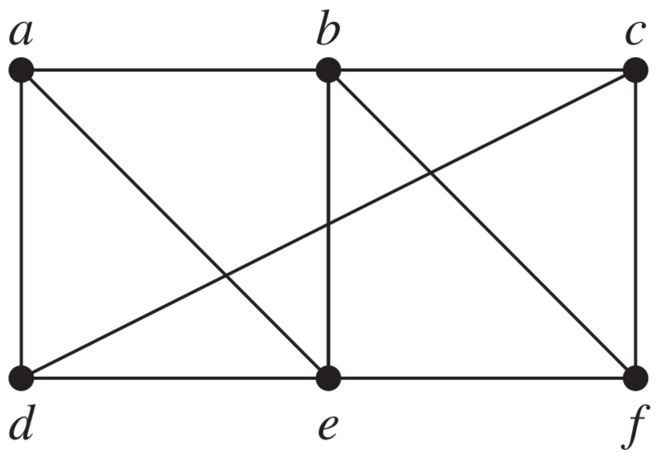
a, d, c, f, e is a simple path of length **4**, because $\{a, d\}$, $\{d, c\}$, $\{c, f\}$, and $\{f, e\}$ are all edges.



Connectivity (5/20)

Example 2:

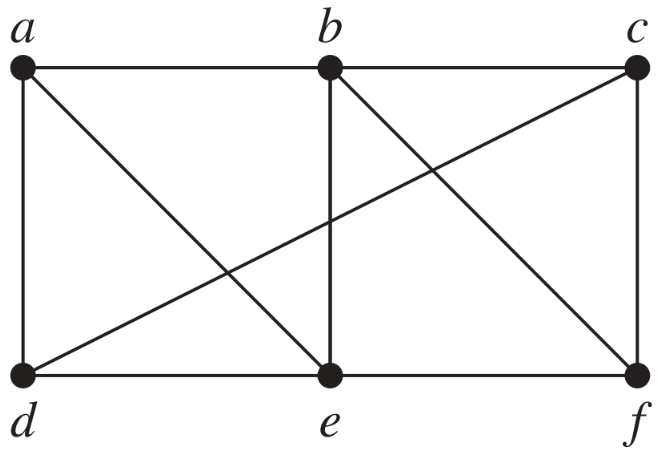
d, e, c, a is **not** a path, because $\{e, c\}$ is not an edge.



Connectivity (6/20)

Example 3:

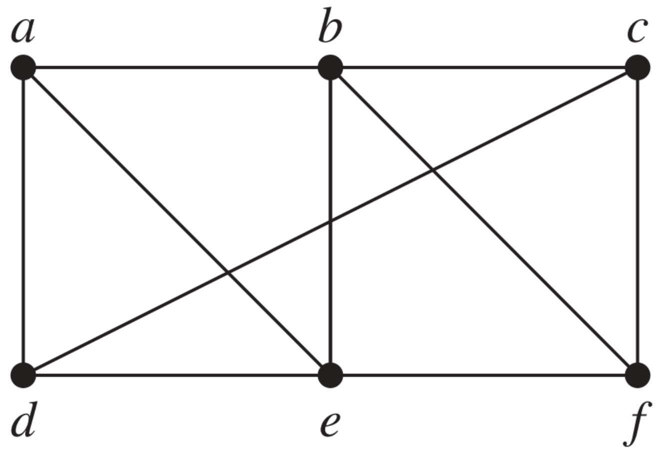
b, c, f, e, b is a circuit of length **4**, because $\{b, c\}$, $\{c, f\}$, $\{f, e\}$, and $\{e, b\}$ are all edges, and this path begins and ends at b.



Connectivity (7/20)

Example 4:

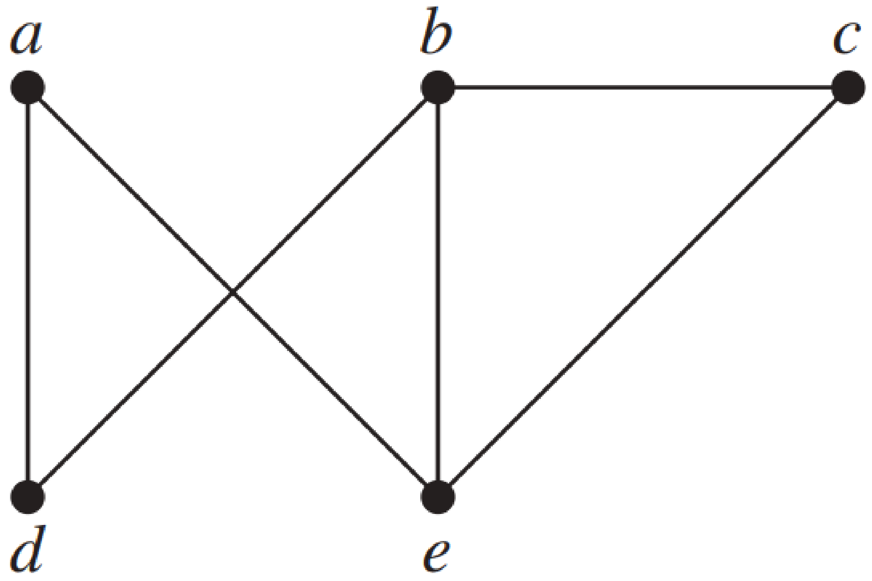
The path $a, b, e, d, a, b,$ which is of length **5**, is not simple because it contains the edge $\{a, b\}$ twice.



Connectivity (8/20)

Example 5 (1/4):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



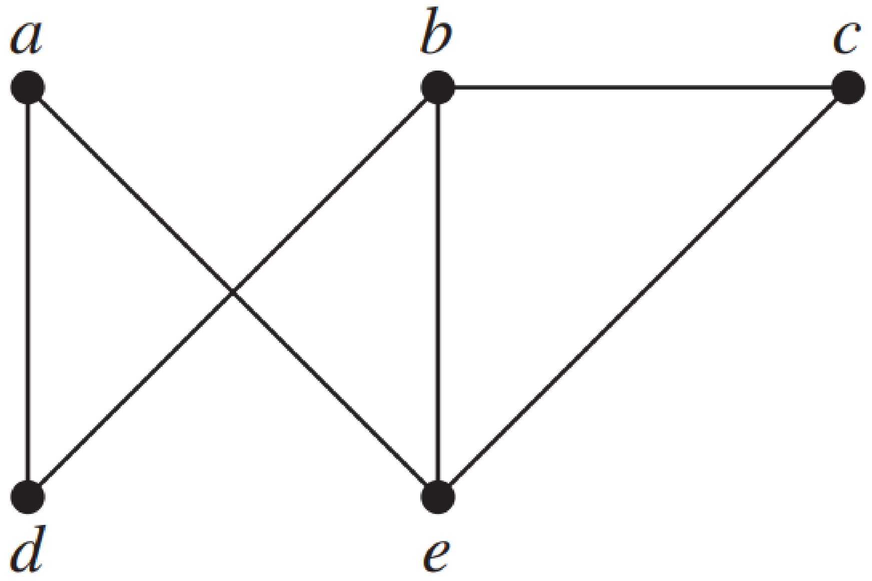
a) a, e, b, c, b

1. Path (---)
2. Simple (---)
3. Circuit (---)
4. Length (---)

Connectivity (8/20)

Example 5 (1/4):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



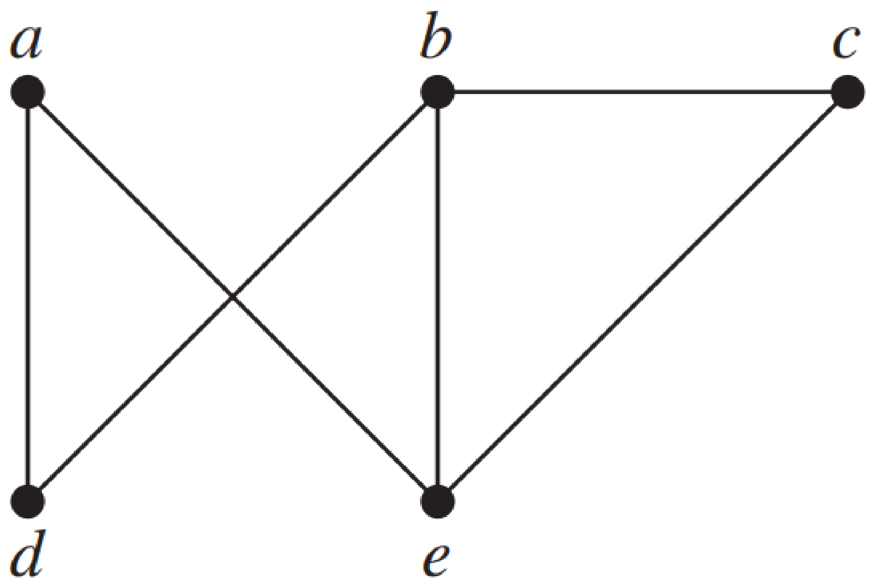
a) a, e, b, c, b

1. Path (Yes)
2. Simple (No)
3. Circuit (No)
4. Length (4)

Connectivity (8/20)

Example 5 (2/4):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



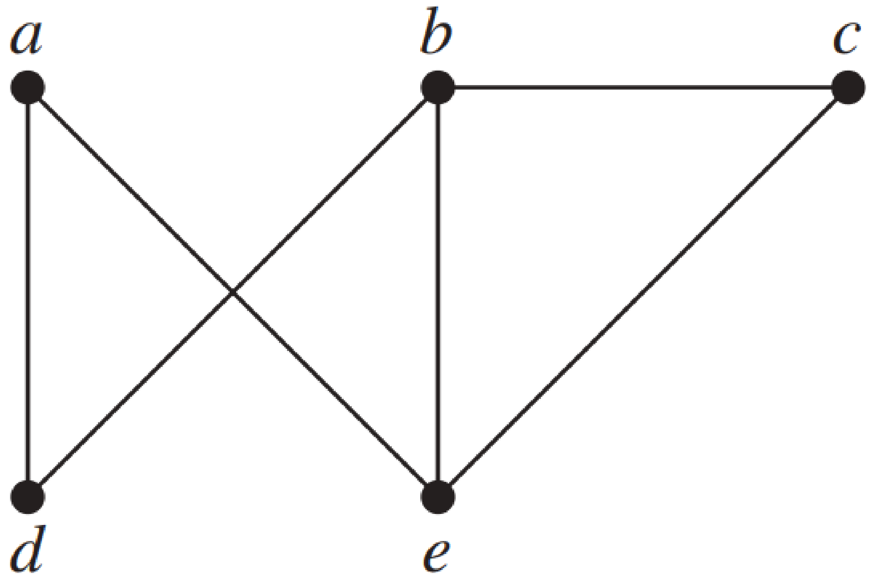
b) a, e, a, d, b, c, a

1. Path (---)
2. Simple (---)
3. Circuit (---)
4. Length (---)

Connectivity (8/20)

Example 5 (2/4):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



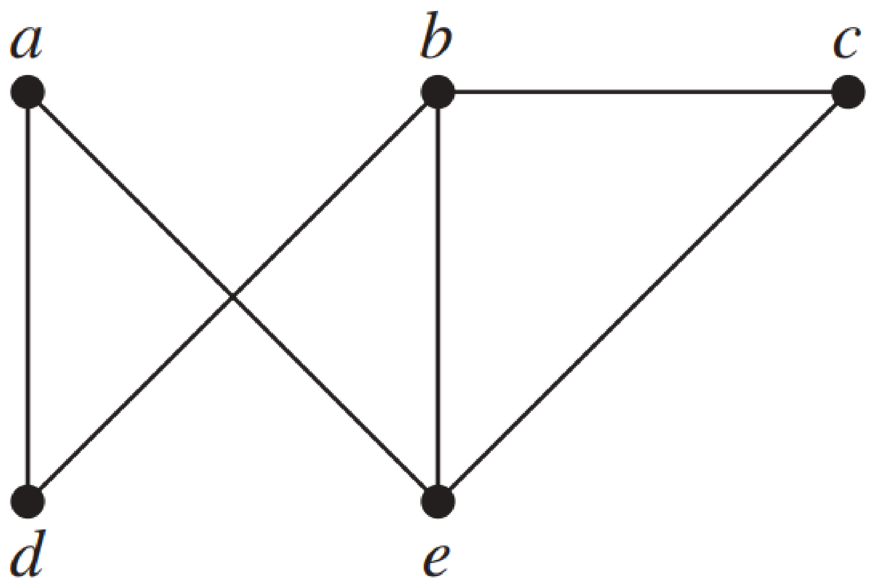
b) a, e, a, d, b, c, a

1. Path (No) $\{c, a\}$ not edge
2. ~~Simple (---)~~
3. ~~Circuit (---)~~
4. ~~Length (---)~~

Connectivity (8/20)

Example 5 (3/4):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



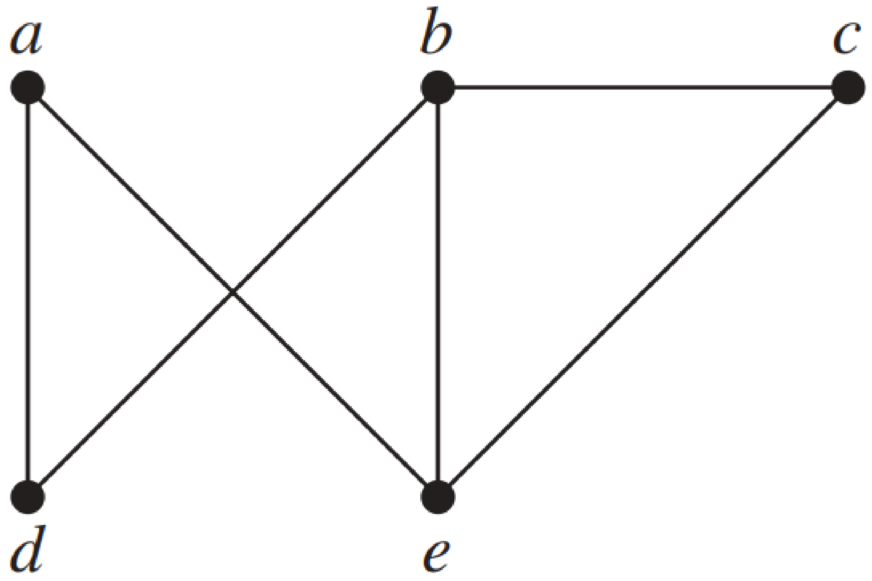
c) e, b, a, d, b, e

1. Path (---)
2. Simple (---)
3. Circuit (---)
4. Length (---)

Connectivity (8/20)

Example 5 (3/4):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



c) e, b, a, d, b, e

1. Path (No) $\{b, a\}$ not edge

~~2. Simple (---)~~

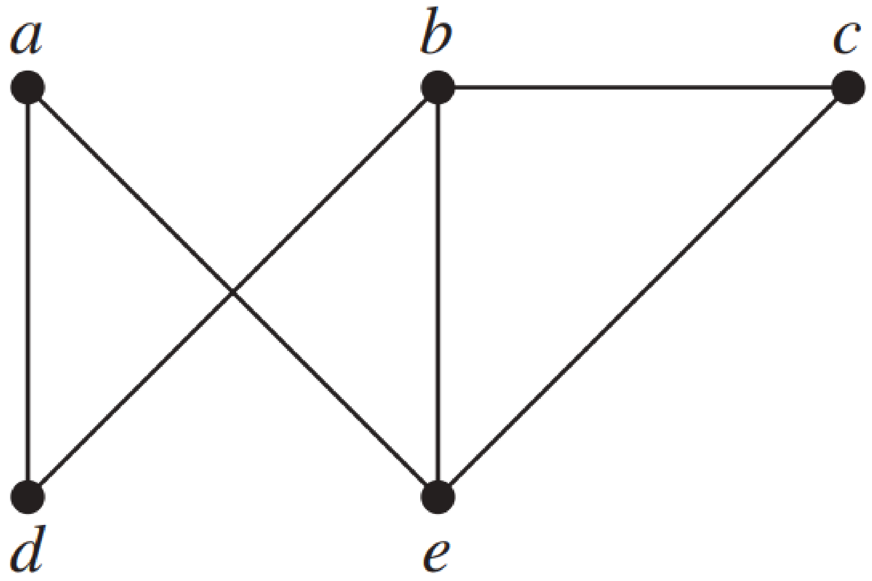
~~3. Circuit (---)~~

~~4. Length (---)~~

Connectivity (8/20)

Example 5 (4/4):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



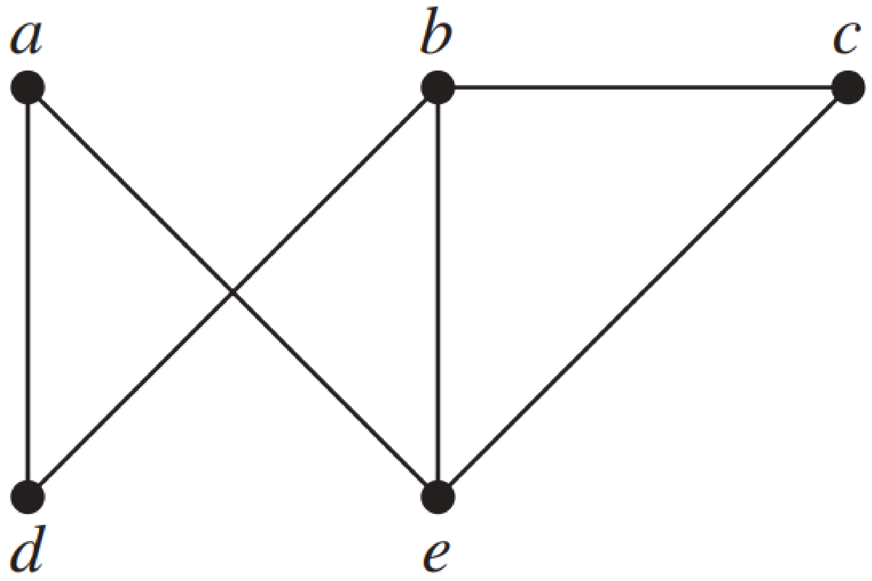
d) c, b, d, a, e, c

1. Path (---)
2. Simple (---)
3. Circuit (---)
4. Length (---)

Connectivity (8/20)

Example 5 (4/4):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



d) c, b, d, a, e, c

1. Path (Yes)
2. Simple (Yes)
3. Circuit (Yes)
4. Length (5)

Connectivity (9/20)

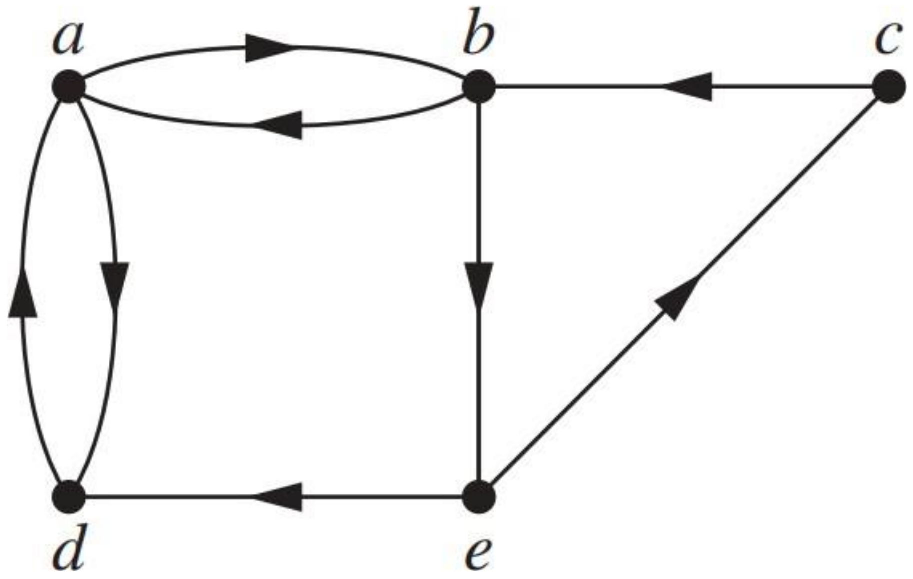
Definition 2:

Let n be a nonnegative integer and G a **directed** graph. A path of length n from u to v in G is a sequence of edges $e_1, e_2, \dots, e_n$ of G such that e_1 is associated with (x_0, x_1), e_2 is associated with (x_1, x_2), and so on, with e_n associated with (x_{n-1}, x_n), where $x_0 = u$ and $x_n = v$. When there are no multiple edges in the directed graph, this path is denoted by its vertex sequence $x_0, x_1, x_2, \dots, x_n$. A path of length greater than zero that begins and ends at the same vertex is called a *circuit* or *cycle*. A path or circuit is called *simple* if it does not contain the same edge more than once.

Connectivity (10/20)

Example 6 (1/3):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



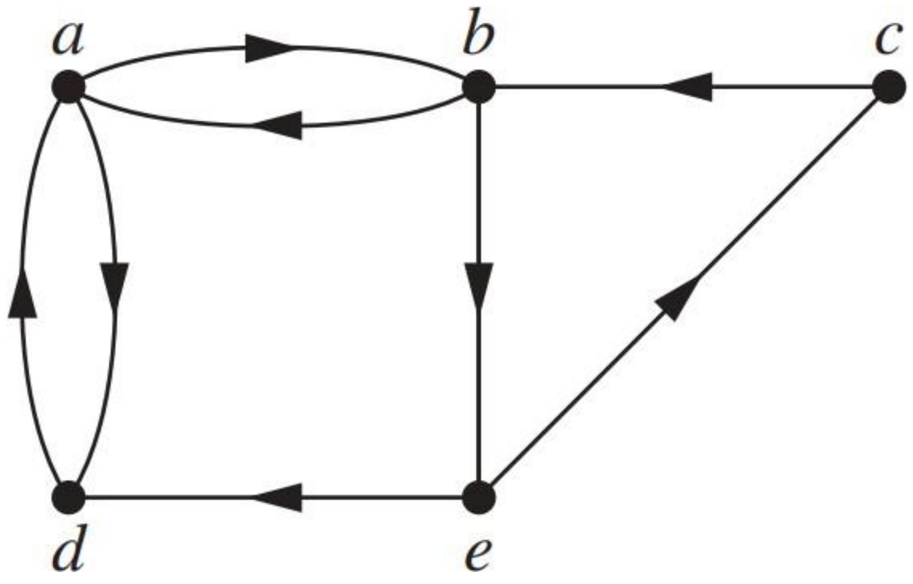
a) a, b, e, c, b

1. Path (---)
2. Simple (---)
3. Circuit (---)
4. Length (---)

Connectivity (10/20)

Example 6 (1/3):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



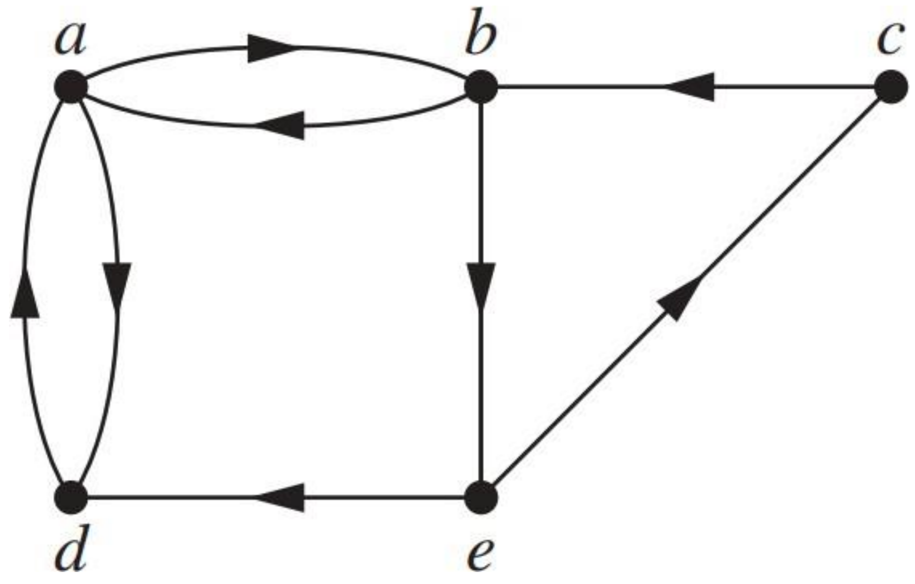
a) a, b, e, c, b

1. Path (Yes)
2. Simple (Yes)
3. Circuit (No)
4. Length (4)

Connectivity (10/20)

Example 6 (2/3):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



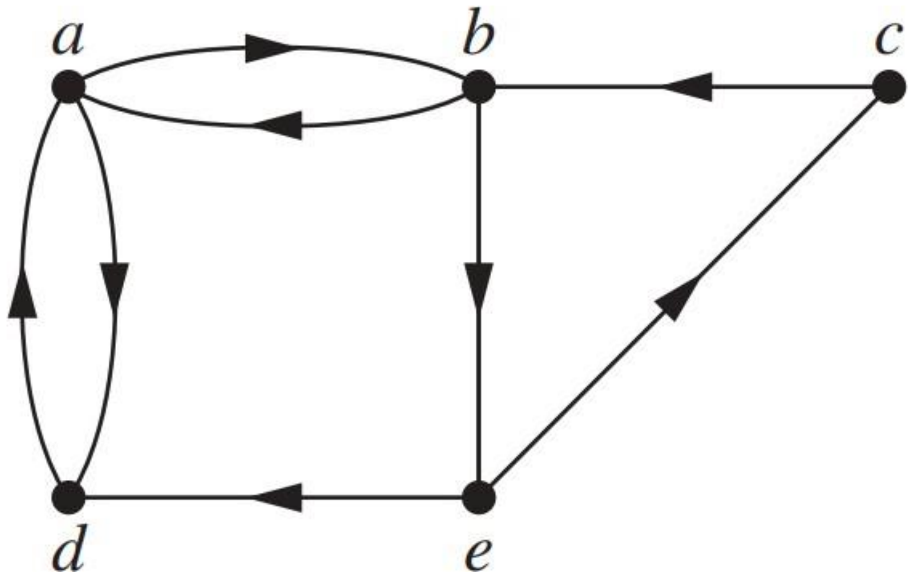
b) a, d, a, d, a

1. Path (---)
2. Simple (---)
3. Circuit (---)
4. Length (---)

Connectivity (10/20)

Example 6 (2/3):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



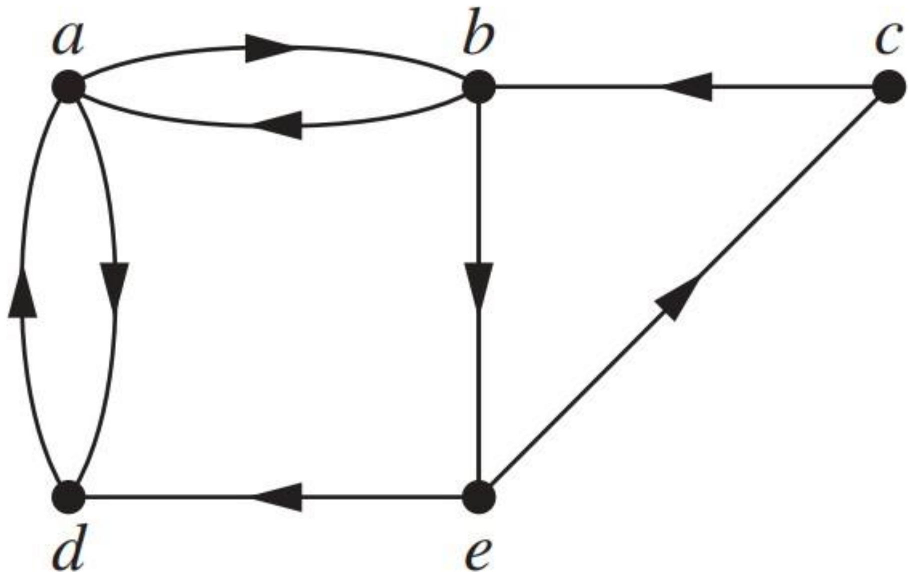
b) a, d, a, d, a

1. Path (Yes)
2. Simple (No)
3. Circuit (Yes)
4. Length (4)

Connectivity (10/20)

Example 6 (3/3):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



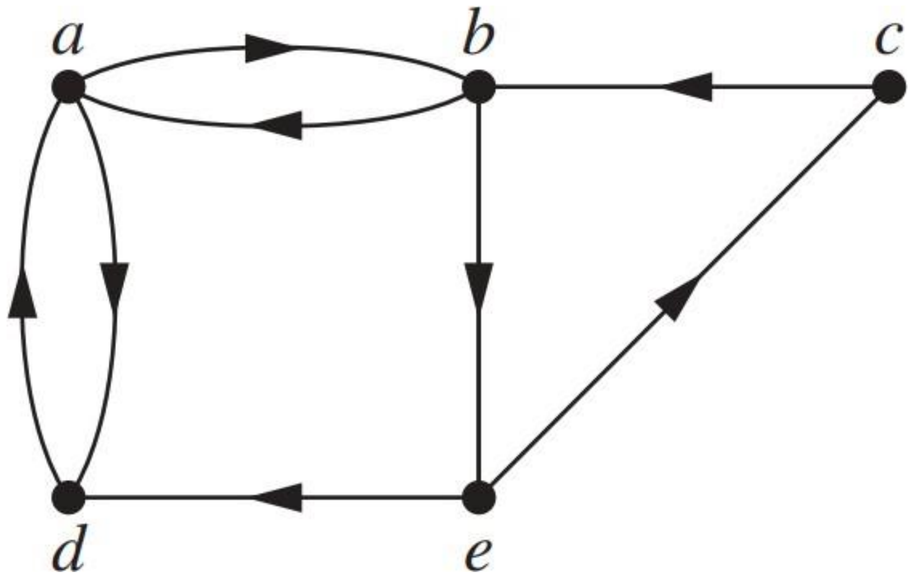
c) a, d, b, e, a

1. Path (---)
2. Simple (---)
3. Circuit (---)
4. Length (---)

Connectivity (10/20)

Example 6 (3/3):

Does each of these lists of vertices form a path in the following graph? Which paths are simple? Which are circuits? What are the lengths of those that are paths?



c) a, d, b, e, a

1. Path (No) $\{d, b\}$ not edge

~~2. Simple (---)~~

~~3. Circuit (---)~~

~~4. Length (---)~~

Connectivity (11/20)

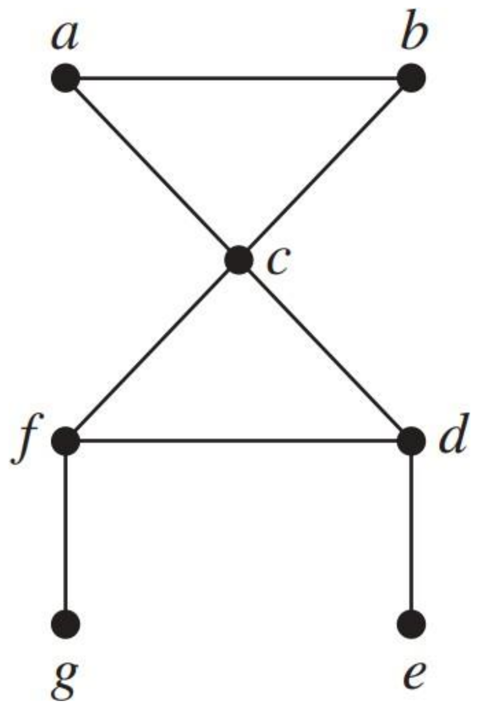
Definition 3:

An undirected graph is called *connected* if there is a path between every pair of distinct vertices of the graph. An undirected graph that is not connected is called *disconnected*. We say that we *disconnect* a graph when we remove vertices or edges, or both, to produce a disconnected subgraph.

Connectivity (12/20)

Example 7 (1/2):

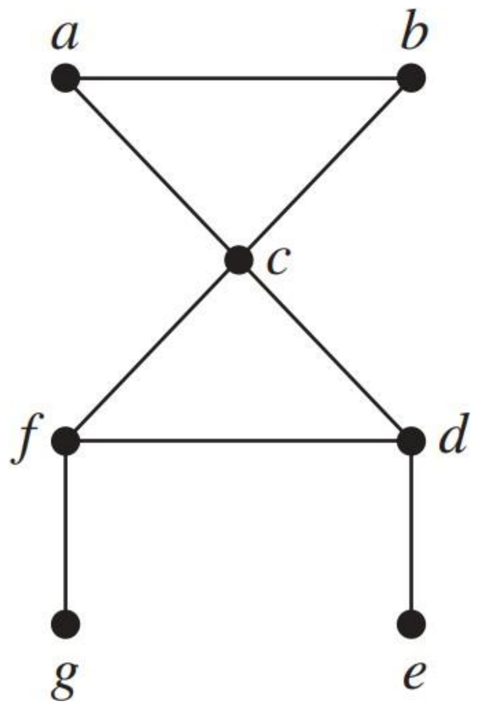
Determine whether the given graph is connected



Connectivity (12/20)

Example 7 (1/2):

Determine whether the given graph is connected

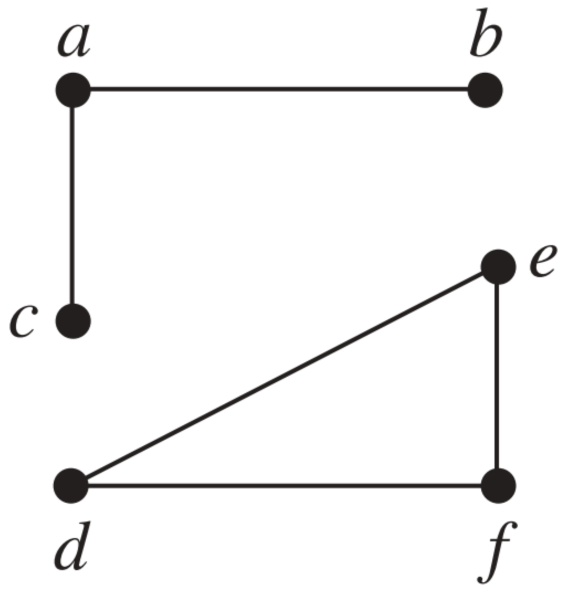


Connected: because there is a path between every pair of distinct vertices of the graph.

Connectivity (12/20)

Example 7 (2/2):

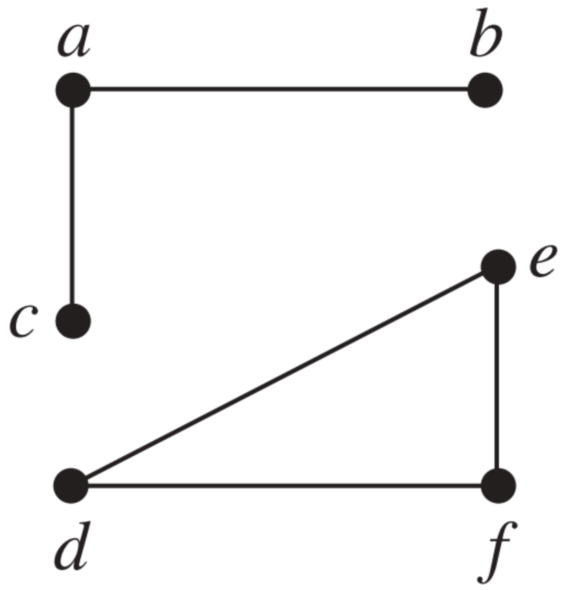
Determine whether the given graph is connected



Connectivity (12/20)

Example 7 (2/2):

Determine whether the given graph is connected



Disconnected: For instance, there is no path between vertices a and d.

Connectivity (13/20)

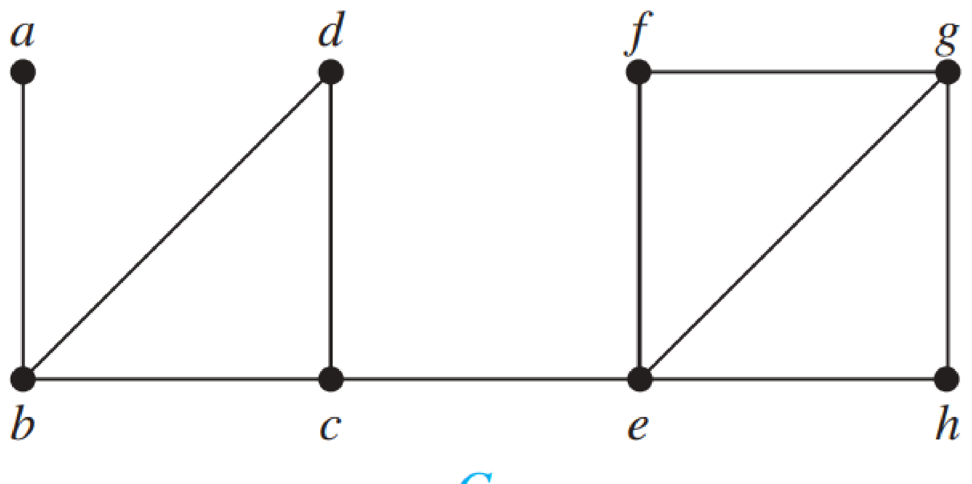
Vertex Connectivity and Edge Connectivity:

Sometimes the removal from a graph of a vertex and all incident edges produces a subgraph with more connected components. Such vertices are called *cut vertices*. We define the *vertex connectivity* of a noncomplete graph G, denoted by $\kappa(G)$, as the minimum number of vertices in a vertex cut.

If a graph has a *cut edge*, then we need only remove it to disconnect G. The *edge connectivity* of a graph G, denoted by $\lambda(G)$, is the minimum number of edges in an edge cut of G.

Connectivity (14/20)

Example 8 (1/2):



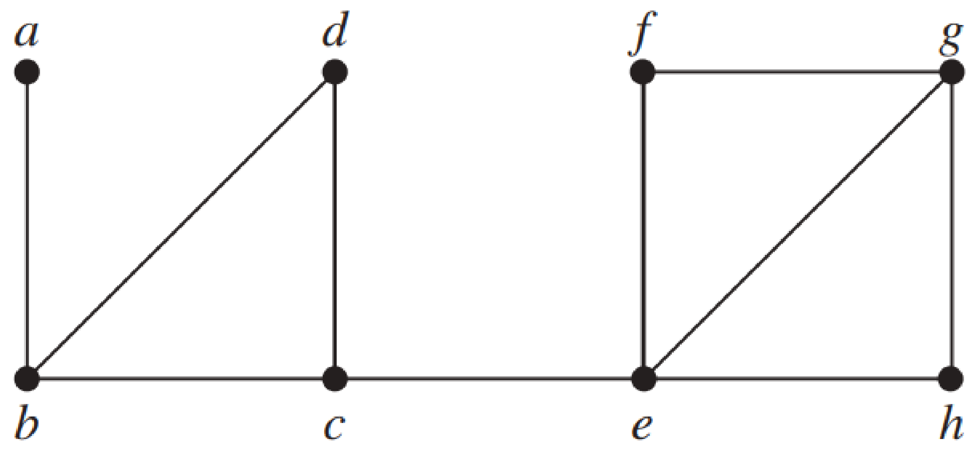
G_1

For instance, ***c*** is a cut vertex of G_1. Also, ***b*** and ***e*** are cut vertices of G_1

Therefore, the vertex connectivity of G_1, $\kappa(G_1) = 1$

Connectivity (14/20)

Example 8 (2/2):



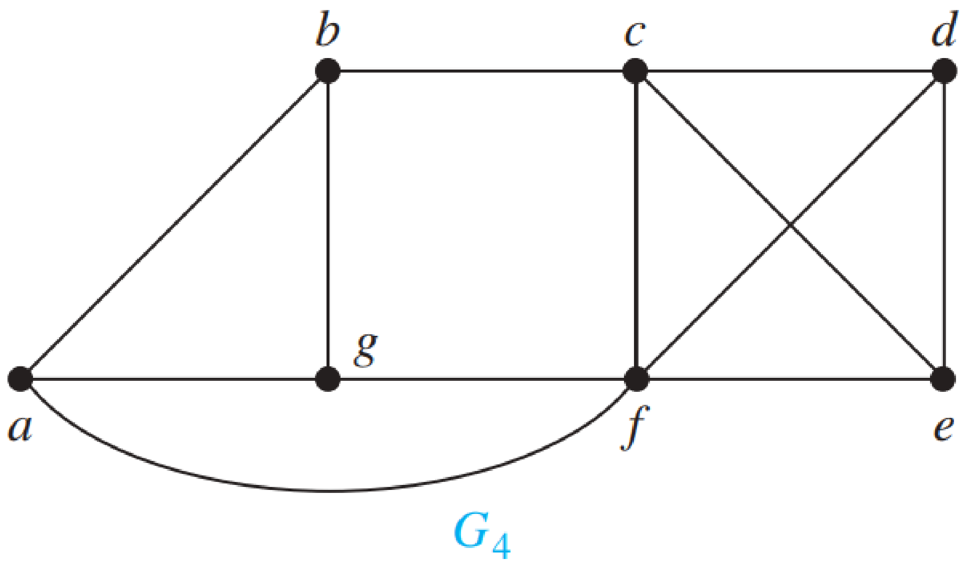
G_1

For instance, $\{\boldsymbol{c}, \boldsymbol{e}\}$ is a cut edge of G_1. Also, $\{\boldsymbol{a}, \boldsymbol{b}\}$ is a cut edge of G_1

Therefore, the edge connectivity of G_1, $\lambda(G_1) = 1$

Connectivity (15/20)

Example 9 (1/2):

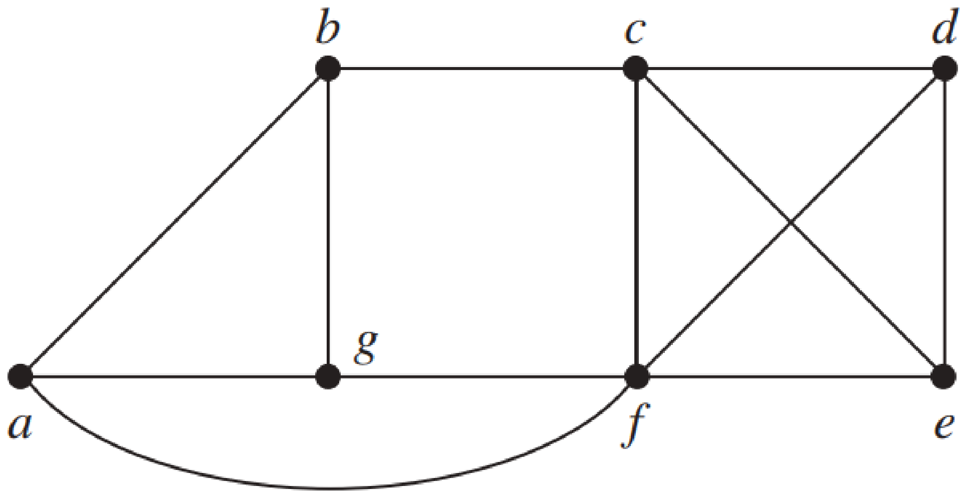


G_4 has a vertex cut of size two, $\{c, f\}$. Also, $\{f, b\}$, ...

Therefore, the vertex connectivity of G_4, $\kappa(G_4) = 2$

Connectivity (15/20)

Example 9 (2/2):



G_4

The removal of the three edges $\{b, c\}$, $\{a, f\}$, and $\{f, g\}$ disconnects the graph G_4.

Therefore, the edge connectivity of G_4, $\lambda(G_4) = 3$

Connectivity (16/20)

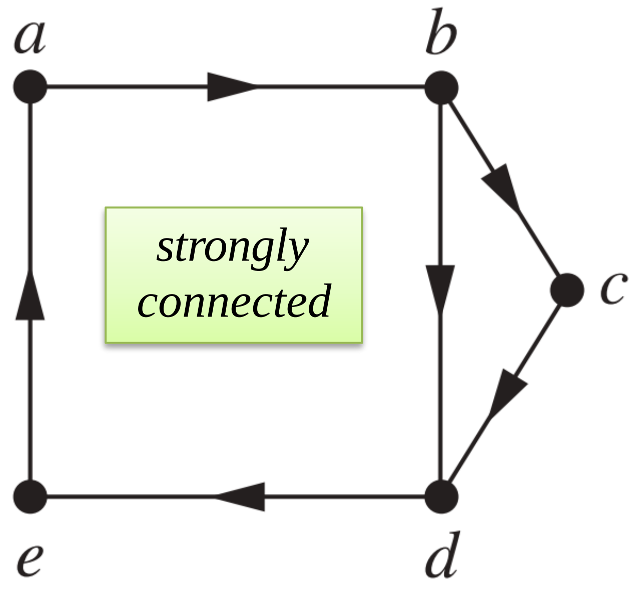
Note:

In a graph representing a computer network, a cut vertex and a cut edge represent an *essential router* and an *essential link* that cannot fail for all computers to be able to communicate.

Connectivity (17/20)

Definition 4:

A directed graph is *strongly connected* if there is a path from a to b and from b to a whenever a and b are vertices in the graph.



Paths: from a to b and from b to a

Paths: from a to c and from c to a

Paths: from a to d and from d to a

.

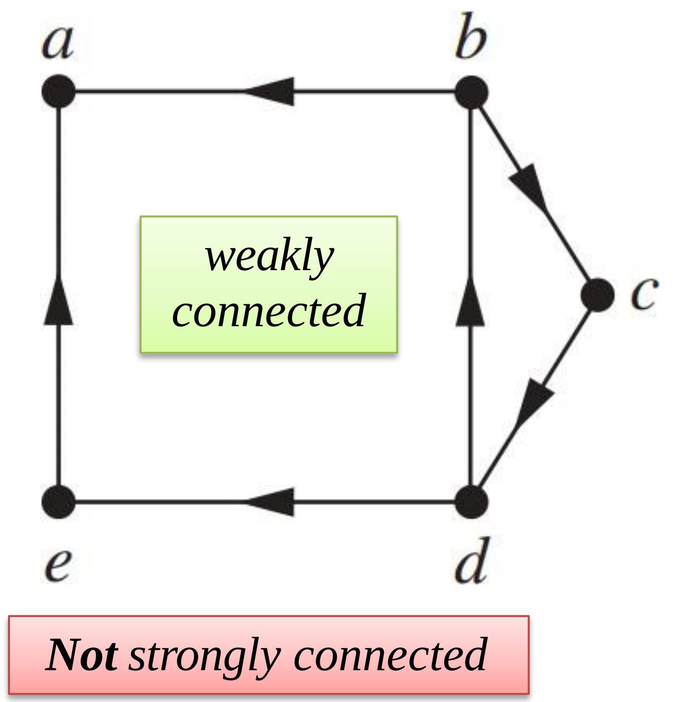
.

.

Connectivity (18/20)

Definition 5:

A directed graph is *weakly connected* if there is a path between every two vertices in the underlying undirected graph.



There is always a path between two vertices *when the directions of the edges are disregarded.* Clearly, any strongly connected directed graph is also weakly connected.

Connectivity (19/20)

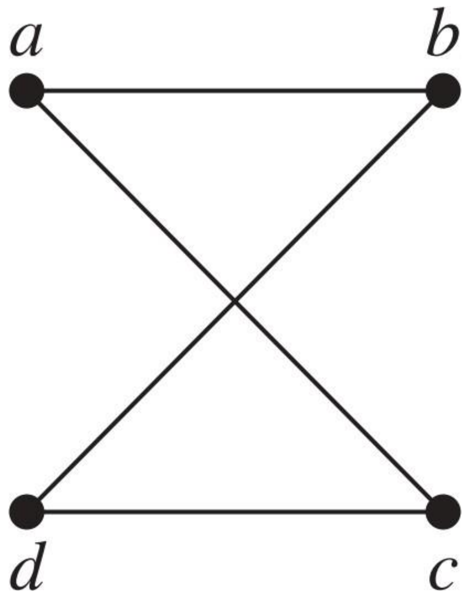
Theorem: Counting Paths Between Vertices:

Let G be a graph with adjacency matrix $\mathbf{A}$ with respect to the ordering $v_1, v_2, \ldots, v_n$ of the vertices of the graph (with directed or undirected edges, with multiple edges and loops allowed). The number of different paths of length r from v_i to v_j, where r is a positive integer, equals the (i, j)th entry of $\mathbf{A}^r$.

Connectivity (20/20)

Example 10 (1/5):

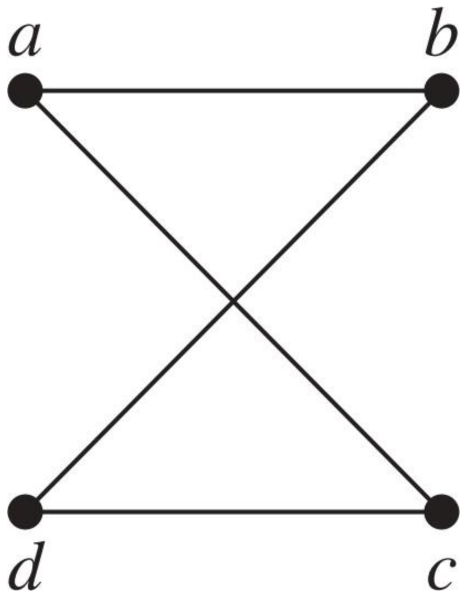
How many paths of length **four** are there from $\boldsymbol{a}$ to $\boldsymbol{d}$ in the following simple graph?



Connectivity (20/20)

Example 10 (2/5):

How many paths of length **four** are there from $\boldsymbol{a}$ to $\boldsymbol{d}$ in the following simple graph?



$$\xrightarrow{\text{adjacency matrix}} \quad \mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Connectivity (20/20)

Example 10 (3/5):

How many paths of length **four** are there from $\boldsymbol{a}$ to $\boldsymbol{d}$ in the following simple graph?

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \xrightarrow{\mathbf{A}^4} \mathbf{A}^4 = \begin{bmatrix} 8 & 0 & 0 & 8 \\ 0 & 8 & 8 & 0 \\ 0 & 8 & 8 & 0 \\ 8 & 0 & 0 & 8 \end{bmatrix}$$

Hence, the number of paths of length four from a to d is the (1, 4)th entry of $\mathbf{A}^4$

Connectivity (20/20)

Example 10 (4/5):

How many paths of length **four** are there from ***a*** to ***d*** in the following simple graph?

There are exactly **eight** paths of length four from a to d.

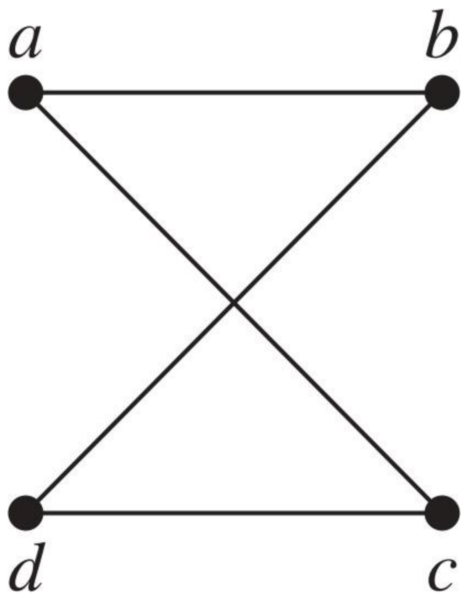
$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \xrightarrow{\mathbf{A}^4} \mathbf{A}^4 = \begin{bmatrix} 8 & 0 & 0 & \bigcirc\!\!\!\!8 \\ 0 & 8 & 8 & 0 \\ 0 & 8 & 8 & 0 \\ 8 & 0 & 0 & 8 \end{bmatrix}$$

Hence, the number of paths of length four from a to d is the (1, 4)th entry of $\mathbf{A}^4$

Connectivity (20/20)

Example 10 (5/5):

How many paths of length **four** are there from $\boldsymbol{a}$ to $\boldsymbol{d}$ in the following simple graph?



Path1: a, b, a, b, d;

Path2: a, b, a, c, d;

Path3: a, b, d, b, d;

Path4: a, b, d, c, d;

Path5: a, c, a, b, d;

Path6: a, c, a, c, d;

Path7: a, c, d, b, d;

Path8: a, c, d, c, d.