DISCRETE STRUCTURES

COURSE INSTRUCTOR: MUHAMMAD SAIF UL ISLAM

Course Outline

- **Logic and Proofs** (Chapter 1)
- **Sets and Functions** (Chapter 2)
- **Relations** (Chapter 9)
- **Number Theory** (Chapter 4)
- Combinatorics and Recurrence
- **Graphs** (Chapter 10)
- Trees
- Discrete Probability

Lecture Outline

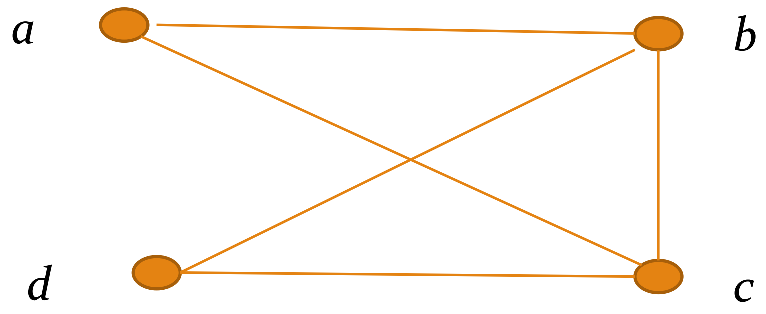
- Graphs and Graph Models
- Graph Terminology and Special Types of Graphs
- Representing Graphs and Graph Isomorphism
- Connectivity
- Euler and Hamiltonian Graphs
- Shortest-Path Problems
- Planar Graphs
- Graph Coloring

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*. An edge is said to *connect* its endpoints.

Example:

This is a graph with four vertices and five edges.

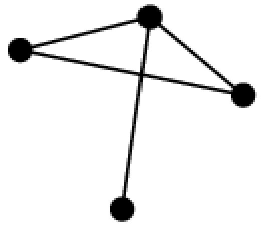


Remarks:

- The graphs we study here are unrelated to graphs of functions studied in Chapter 2.
- We have a lot of freedom when we draw a picture of a graph. All that matters is the connections made by the edges, not the particular geometry depicted. For example, the lengths of edges, whether edges cross, how vertices are depicted, and so on, do not matter
- A graph with an infinite vertex set is called an *infinite graph.* A graph with a finite vertex set is called a ***finite*** *graph*. We (following the text) restrict our attention to finite graphs.

Some Terminology

In a ***simple*** *graph* each edge connects two different vertices and no two edges connect the same pair of vertices.

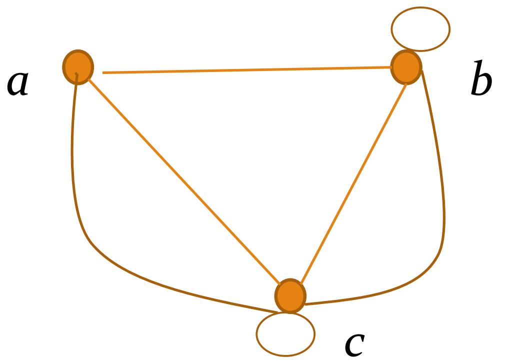


Multigraphs may have multiple edges connecting the same two vertices. When *m* different edges connect the vertices *u* and *v*, we say that {*u*,*v*} is an edge of *multiplicity m*.

An edge that connects a vertex to itself is called a ***loop***.

A ***pseudograph*** may include loops, as well as multiple edges connecting the same pair of vertices.

Example:
This pseudograph has both multiple edges and a loop.

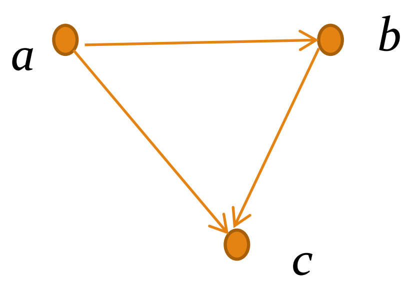


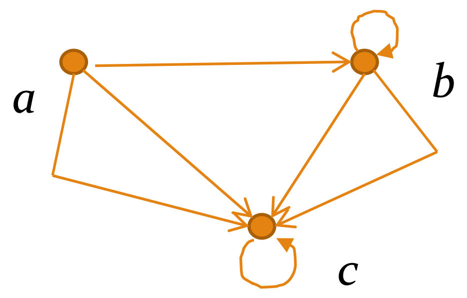
Remark: There is no standard terminology for graph theory. So, it is crucial that you understand the terminology being used whenever you read material about graphs.

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*.

- Graphs where the end points of an edge are not ordered are said to be ***undirected*** *graphs*.
- A ***simple directed graph*** has no loops and no multiple edges.
- A ***directed multigraph*** may have multiple directed edges. When there are m directed edges from the vertex u to the vertex v, we say that (u,v) is an edge of *multiplicity m*.



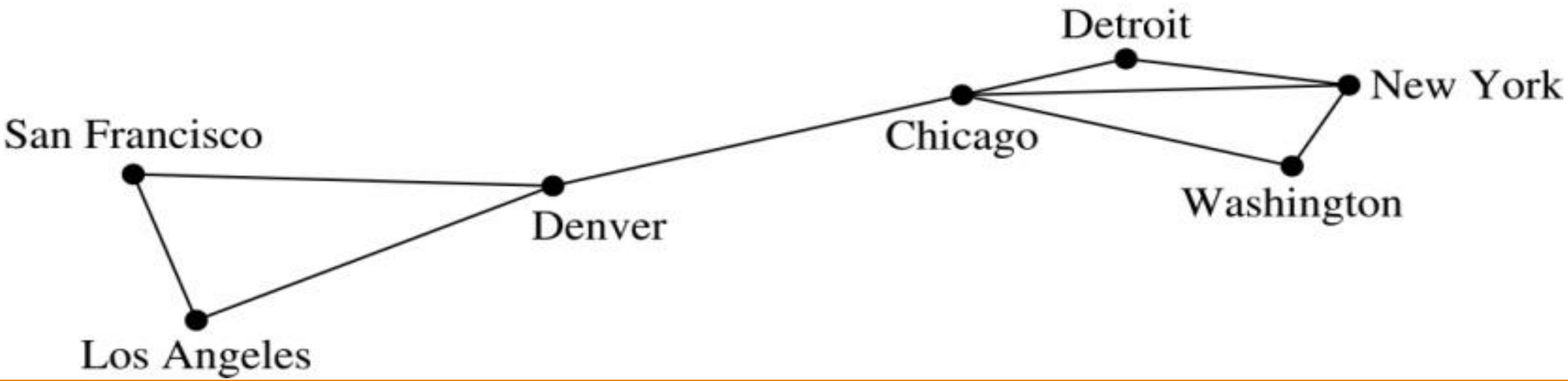


Graph Models:
Computer Networks

When we build a graph model, we use the appropriate type of graph to capture the important features of the application.

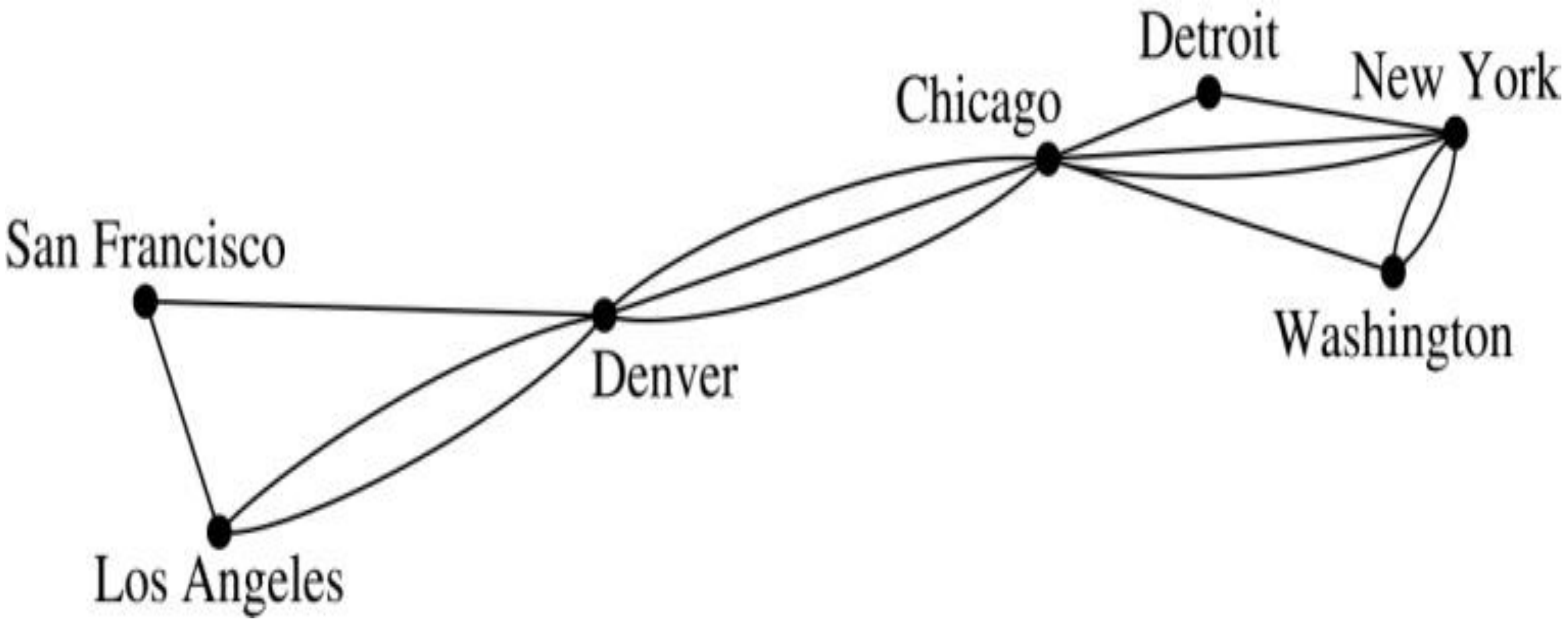
We illustrate this process using graph models of different types of computer networks. In all these graph models, the vertices represent data centers and the edges represent communication links.

To model a computer network where we are only concerned whether two data centers are connected by a communications link, we use a simple graph. This is the appropriate type of graph when we only care whether two data centers are directly linked (and not how many links there may be) and all communications links work in both directions.



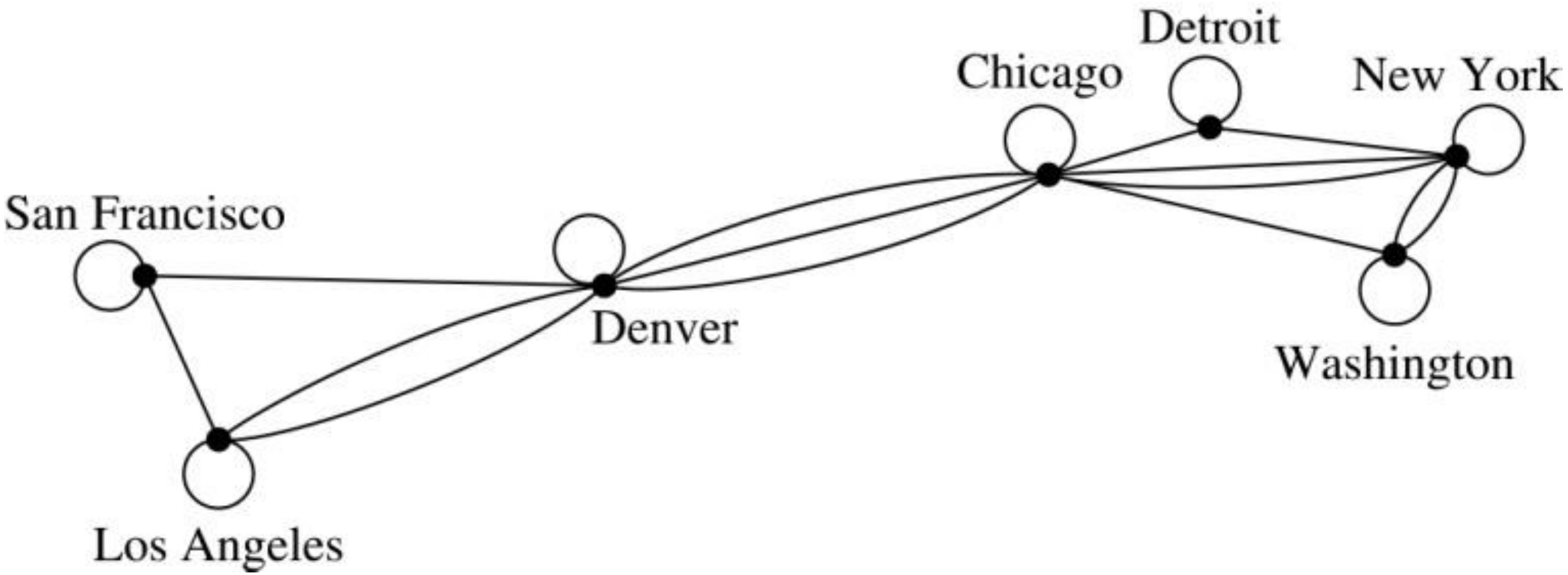
Graph Models:
Computer Networks (*continued*)

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



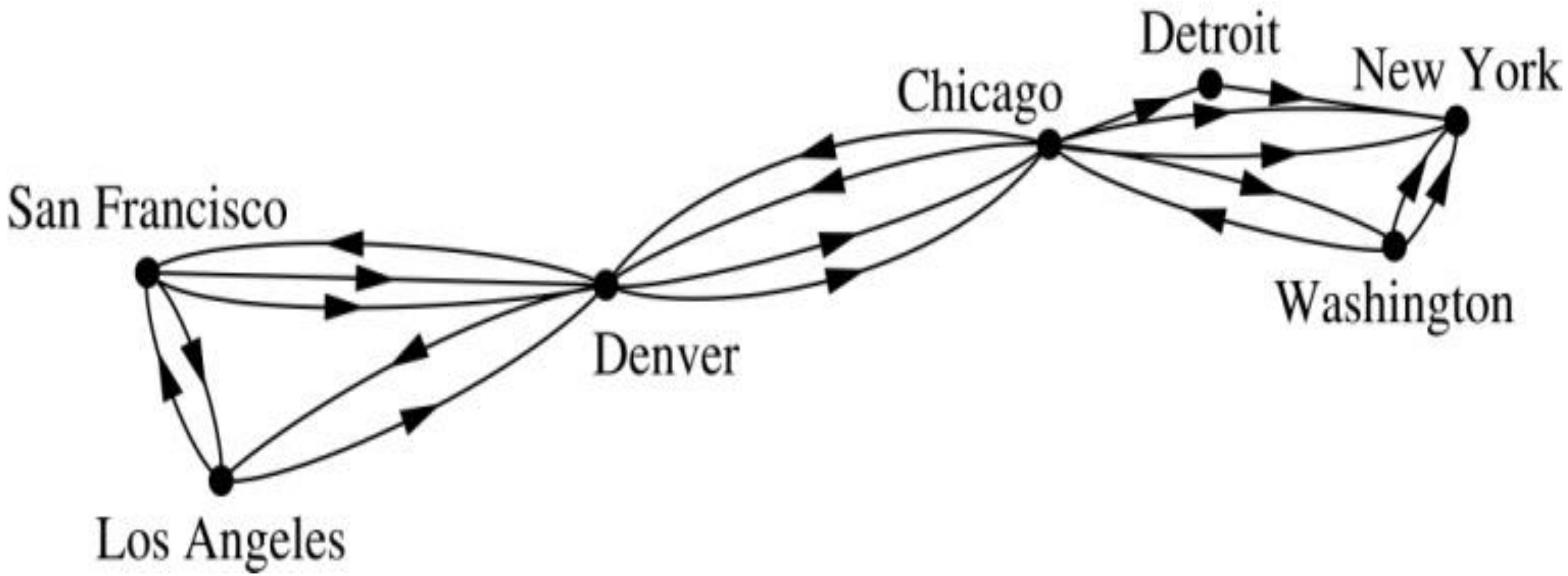
Graph Models: Computer Networks

To model a computer network with diagnostic links at data centers, we use a pseudograph, as loops are needed.



Graph Models: Computer Networks

To model a network with multiple one-way links, we use a directed multigraph. Note that we could use a directed graph without multiple edges if we only care whether there is at least one link from a data center to another data center.



Graph Terminology: Summary

To understand the structure of a graph and to build a graph model, we ask these questions:

- Are the edges of the graph undirected or directed (or both)?
- If the edges are undirected, are multiple edges present that connect the same pair of vertices? If the edges are directed, are multiple directed edges present?
- Are loops present?

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

Other Applications of Graphs

We will illustrate how graph theory can be used in models of:

- Social networks
- Communications networks
- Information networks
- Software design
- Transportation networks
- Biological networks

It’s a challenge to find a subject to which graph theory has not yet been applied.
Can you find an area without applications of graph theory?

Read more about the applications by yourself-->

Graph Models: Social Networks

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

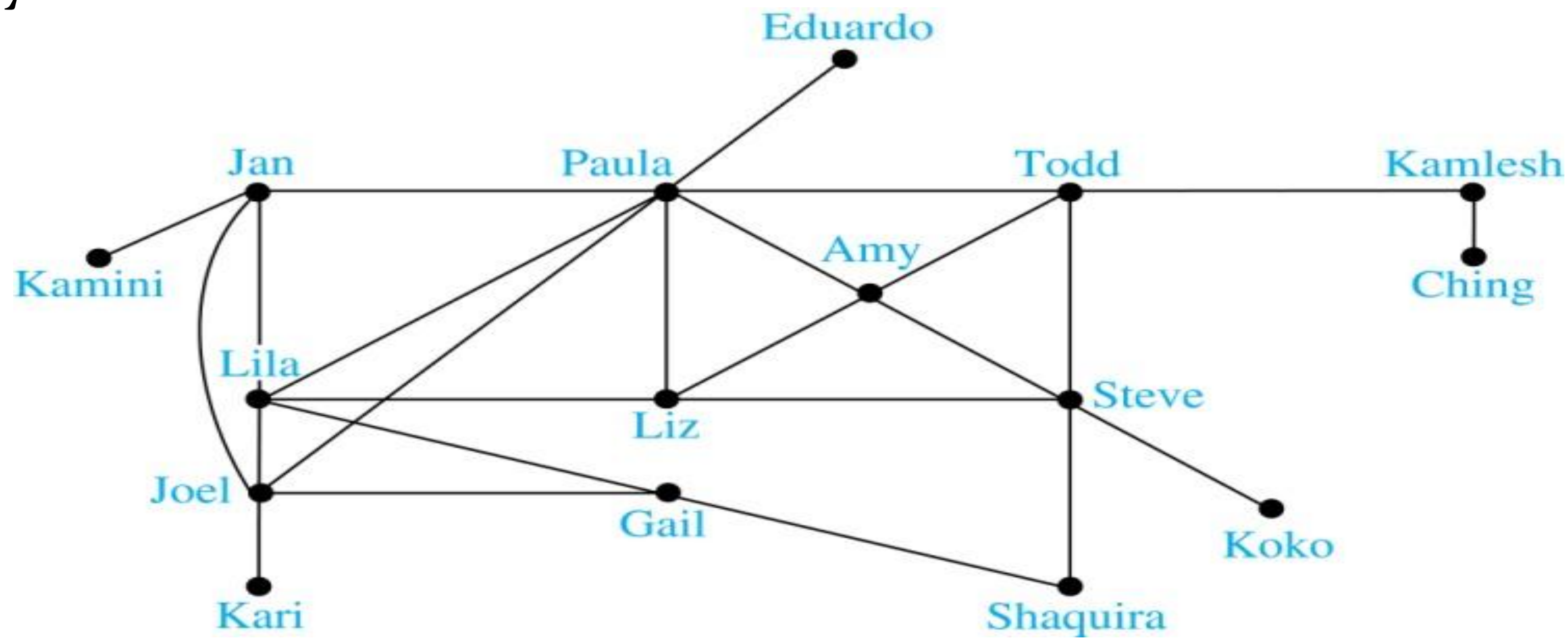
In a *social network*, vertices represent individuals or organizations and edges represent relationships between them.

Useful graph models of social networks include:

- *friendship graphs* - undirected graphs where two people are connected if they are friends (in the real world, on Facebook, or in a particular virtual world, and so on.)
- *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

Graph Models: Social Networks

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



Applications to Information Networks

Graphs can be used to model different types of 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.
- We will explain how the web graph is used by search engines in Section 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.

Transportation Graphs

Graph models are extensively used in the study of transportation networks.

Airline networks can be modeled using directed multigraphs where

- airports are represented by vertices
- each flight is represented by a directed edge from the vertex representing the departure airport to the vertex representing the destination airport

Road networks can be modeled using graphs where

- vertices represent intersections and edges represent roads.
- undirected edges represent two-way roads and directed edges represent one-way roads.

Software Design Applications

Graph models are extensively used in software design. We will introduce two such models here; one representing the dependency between the modules of a software application and the other representing restrictions in the execution of statements in computer programs.

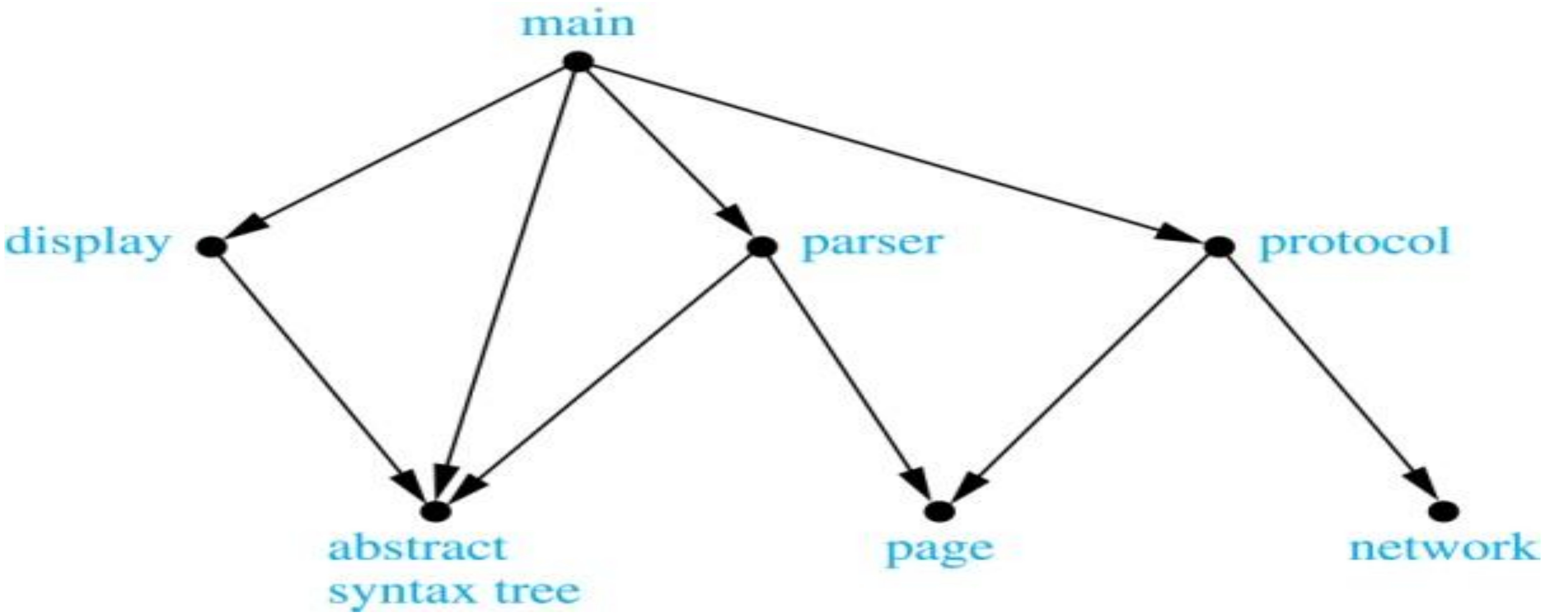
When a top-down approach is used to design software, the system is divided into modules, each performing a specific task.

We use a *module dependency graph* to represent the dependency between these modules. These dependencies need to be understood before coding can be done.

Software Design Applications

- In a module dependency graph vertices represent software modules and there is an edge from one module to another if the second module depends on the first.

Example: The dependencies between the seven modules in the design of a web browser are represented by this module dependency graph.

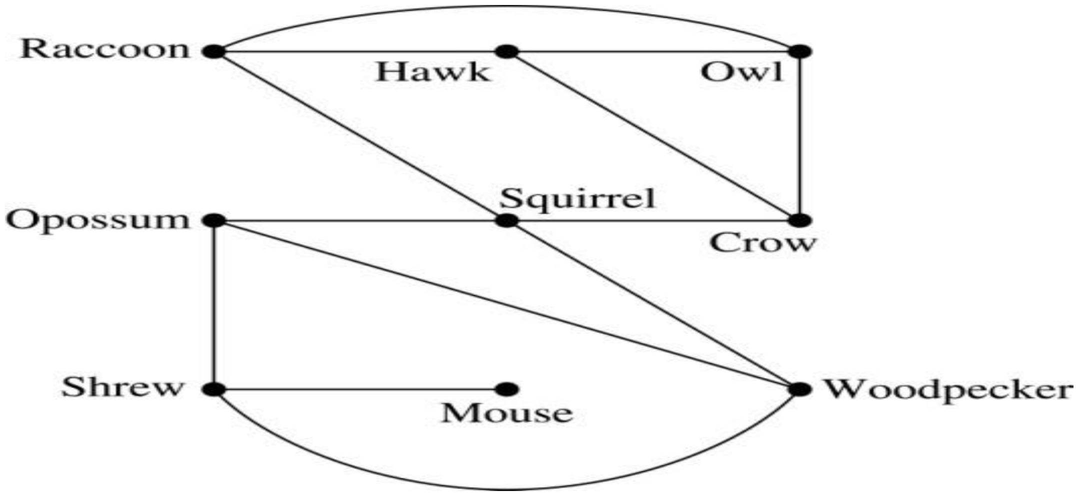


Biological Applications

Graph models are used extensively in many areas of the biological science. We will describe two such models, one to ecology and the other to molecular biology.

Niche overlap graphs model competition between species in an ecosystem

- Vertices represent species and an edge connects two vertices when they represent species who compete for food resources.



Example: This is the niche overlap graph for a forest ecosystem with nine species.

Biological Applications

We can model the interaction of proteins in a cell using a *protein interaction network.*

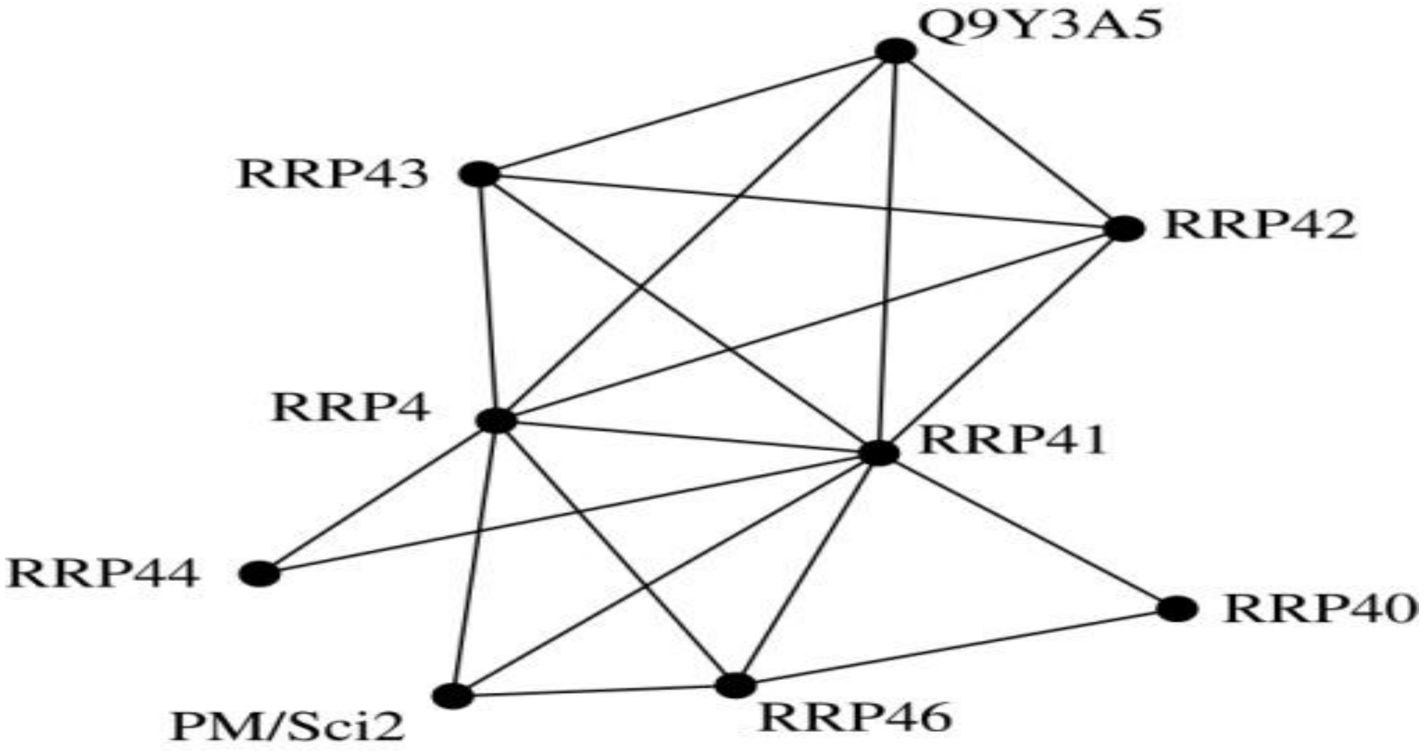
In a *protein interaction graph*, vertices represent proteins and vertices are connected by an edge if the proteins they represent interact.

Protein interaction graphs can be huge and can contain more than 100,000 vertices, each representing a different protein, and more than 1,000,000 edges, each representing an interaction between proteins

Protein interaction graphs are often split into smaller graphs, called *modules*, which represent the interactions between proteins involved in a particular function.

Biological Applications

Example: This is a module of the protein interaction graph of proteins that degrade RNA in a human cell.



Basic Terminology

Definition 1. Two vertices *u*, *v* in an undirected graph *G* are called *adjacent* (or *neighbors*) in *G* if there is an edge *e* between *u* and *v*. Such an edge *e* is called *incident with* the vertices *u* and *v* and *e* is said to *connect* *u* and *v*.

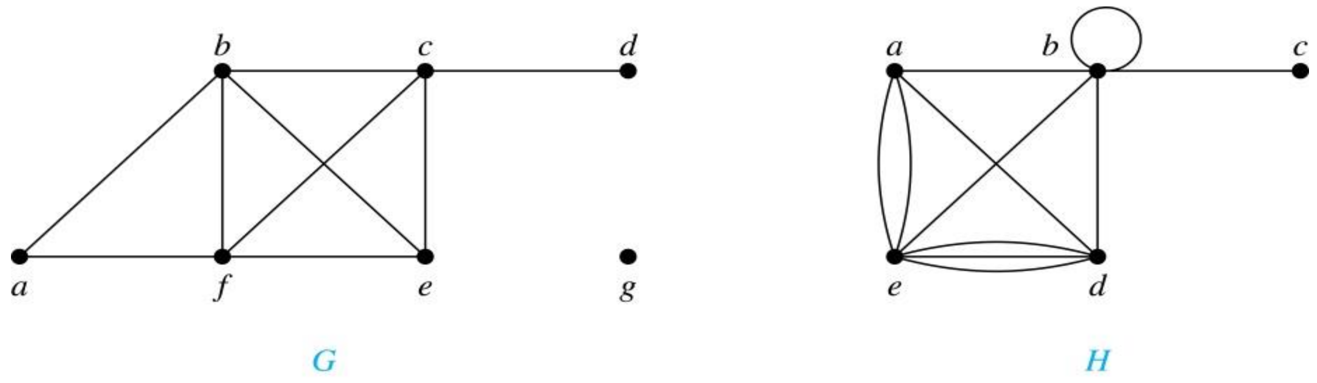
Definition 2. The set of all neighbors of a vertex *v* of *G* = (*V*, *E*), denoted by *N*(*v*), is called the *neighborhood* of *v*. If *A* is a subset of *V*, we denote by *N*(*A*) the set of all vertices in *G* that are adjacent to at least one vertex in *A*. So,

$$N(A) = \bigcup_{v \in A} N(v).$$

Definition 3. The *degree of a vertex in a undirected graph* is the number of edges incident with it, except that a loop at a vertex contributes two to the degree of that vertex. The degree of the vertex *v* is denoted by deg(*v*).

Degrees and Neighborhoods of Vertices

Example: What are the degrees and neighborhoods of the vertices in the graphs G and H?



Solution:

G: **deg(a)** = 2, deg(b) = deg(c) = deg(f) = 4, deg(d) = 1, deg(e) = 3, deg(g) = 0.

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

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

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

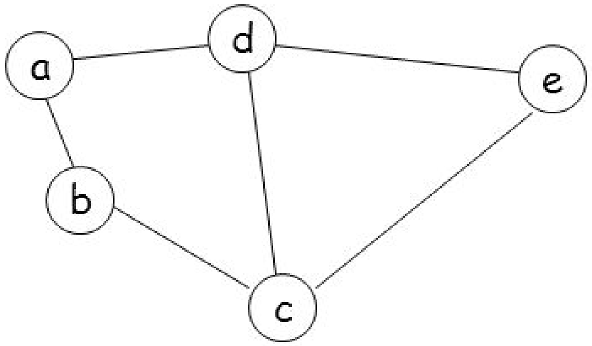
$N(a) = \{b, d, e\}$, $N(b) = \{a, b, c, d, e\}$, $N(c) = \{b\}$, $N(d) = \{a, b, e\}$,

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

Degree of Vertices

Theorem: An undirected graph 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 an undirected graph $G = (V, E)$ with m edges. Then



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

$$2e = \sum_{u \in OddDegVertices} \deg(u) + \sum_{v \in EvenDegVertices} \deg(w)$$

$$2k = \sum_{u \in OddDegVertices} \deg(u)$$

deg(d) = 3 and deg(c) = 3

Handshaking Theorem

Theorem: If G is any graph, then the sum of the degrees of all the vertices of G equals twice the number of edges of G.

Specifically, if the vertices of G are v**1**, v**2**, …, v**n**, where n is a positive integer, then

the total degree of G = deg(v1) + deg(v2) + … + deg(vn) = 2 . (the number of edges of G)

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

COROLLARY: The total degree of G is an even number

Handshaking Theorem

We now give two examples illustrating the usefulness of the handshaking theorem.

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

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

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

Solution: This is not possible by the handshaking thoerem, because the sum of the degrees of the vertices $3 \cdot 5 = 15$ is odd.

Directed Graphs

Recall the definition of a directed graph.

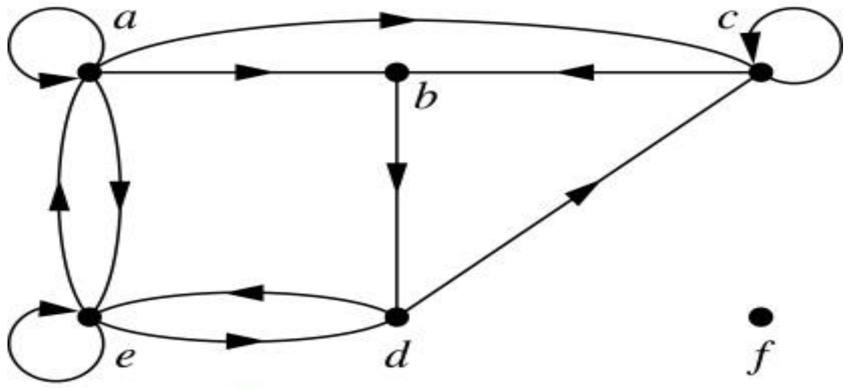
Definition: An *directed graph G =* (*V, E)* consists of *V,* a nonempty set of *vertices* (or *nodes*), and *E,* a set of *directed edges* or *arcs.* Each edge is an ordered pair of vertices. The directed edge (*u*,*v*) is said to start at *u* and end at *v*.

Definition: Let (*u,v*) be an edge in *G*. Then *u* is the *initial vertex* of this edge and is *adjacent to v* and *v* is the *terminal* (or *end*) *vertex* of this edge and is *adjacent from u*. The initial and terminal vertices of a loop are the same.

Directed Graphs (*continued*)

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

Example: In the graph *G* we have



G

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

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

Directed Graphs (*continued*)

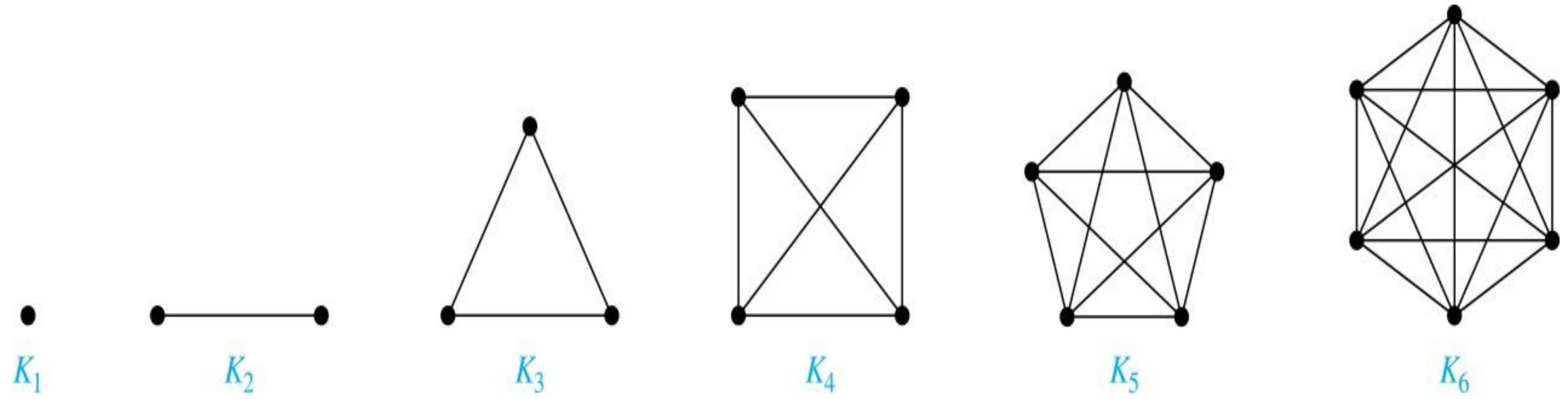
Theorem 3: Let *G* = (*V, E*) be a graph with directed edges. Then:

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

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.

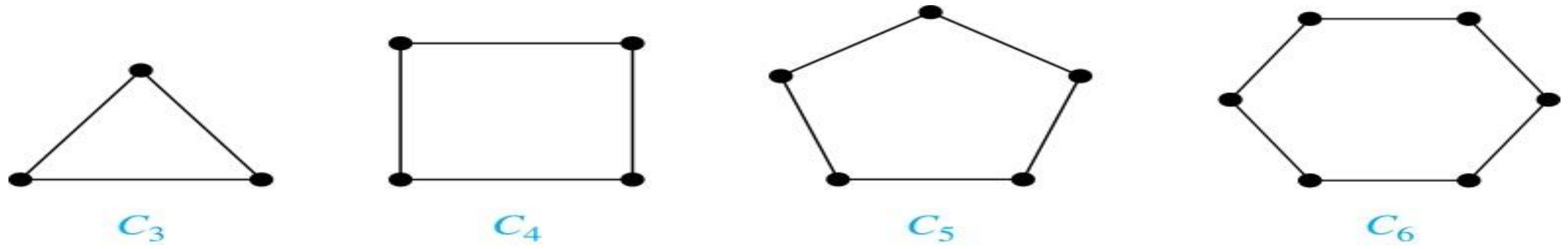
Special Types of Simple Graphs: Complete Graphs

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.

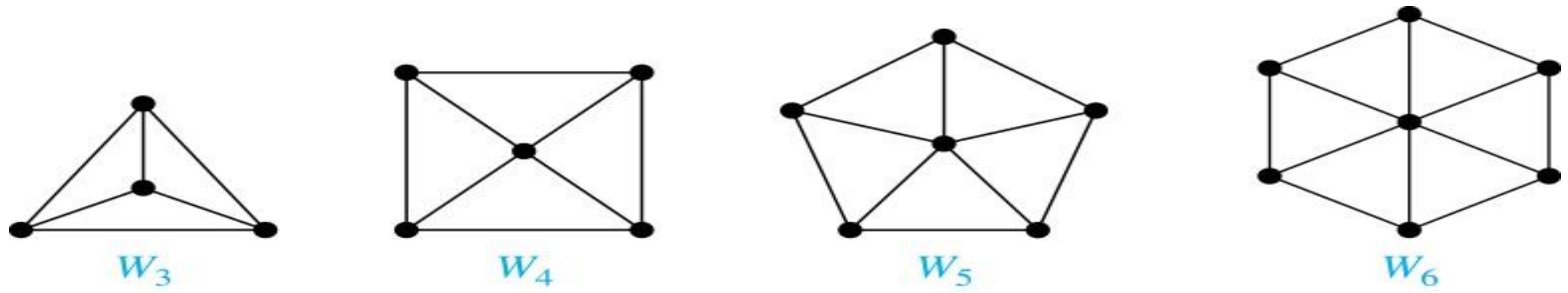


Special Types of Simple Graphs: Cycles and Wheels

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

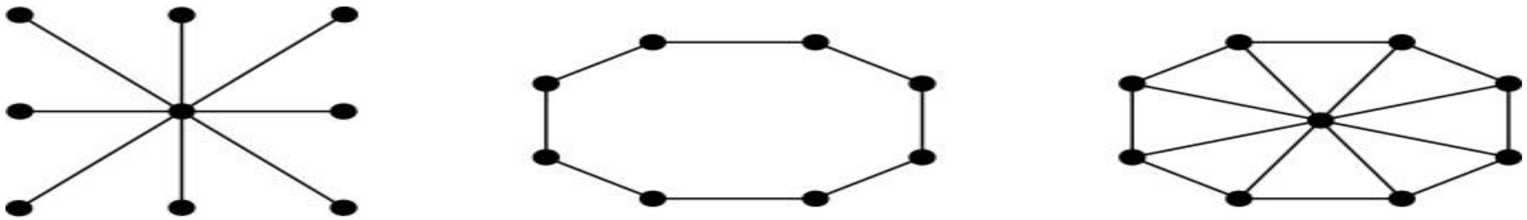


A *wheel* W_n is obtained by adding an additional vertex to a cycle C_n for $n \geq 3$ and connecting this new vertex to each of the n vertices in C_n by new edges.



Special Types of Graphs and Computer Network Architecture

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



(a) (b) (c)

Some local area networks use a *star topology*, which is a complete bipartite graph $K_{1,n}$, as shown in (a). All devices are connected to a central control device.

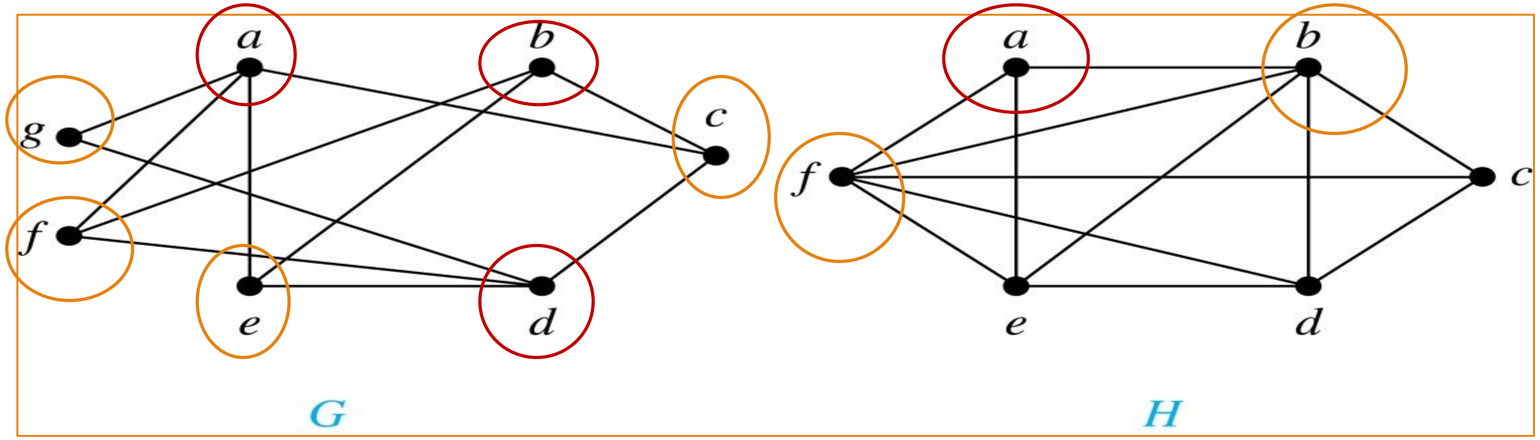
Other local networks are based on a *ring topology*, where each device is connected to exactly two others using C_n, as illustrated in (b). Messages may be sent around the ring.

Others, as illustrated in (c), use a W_n – based topology, combining the features of a star topology and a ring topology.

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.

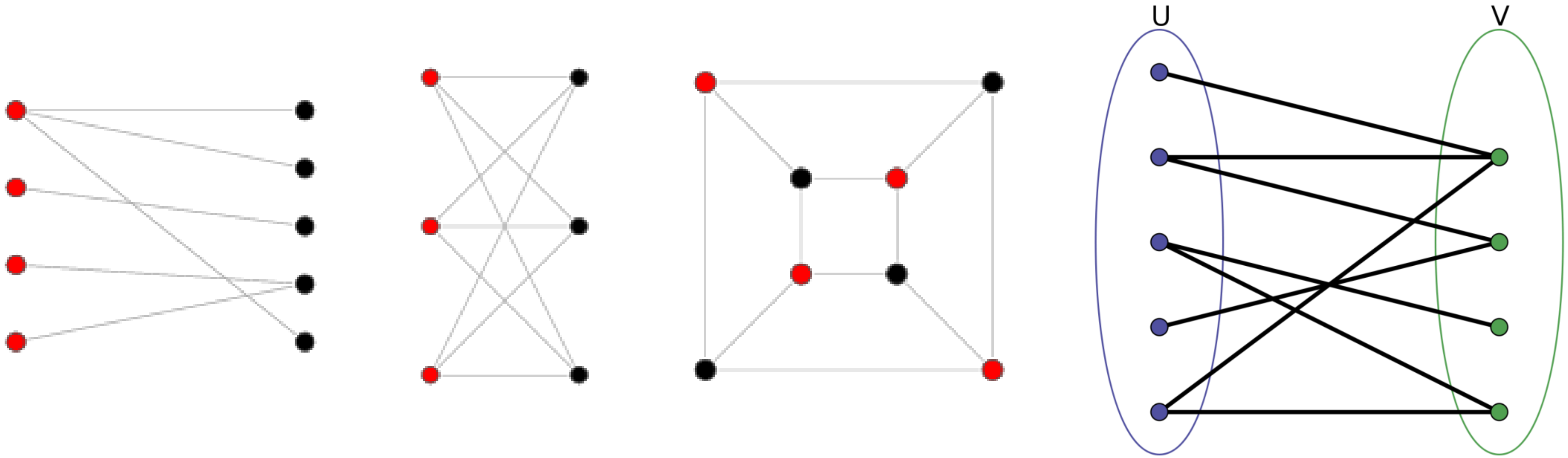
It is not hard to show that an equivalent definition of a bipartite graph is a graph where it is possible to color the vertices red or blue so that no two adjacent vertices are the same color.



G is bipartite

H is not bipartite since if we color a red, then the adjacent vertices f and b must both be blue.

Bipartite Graphs

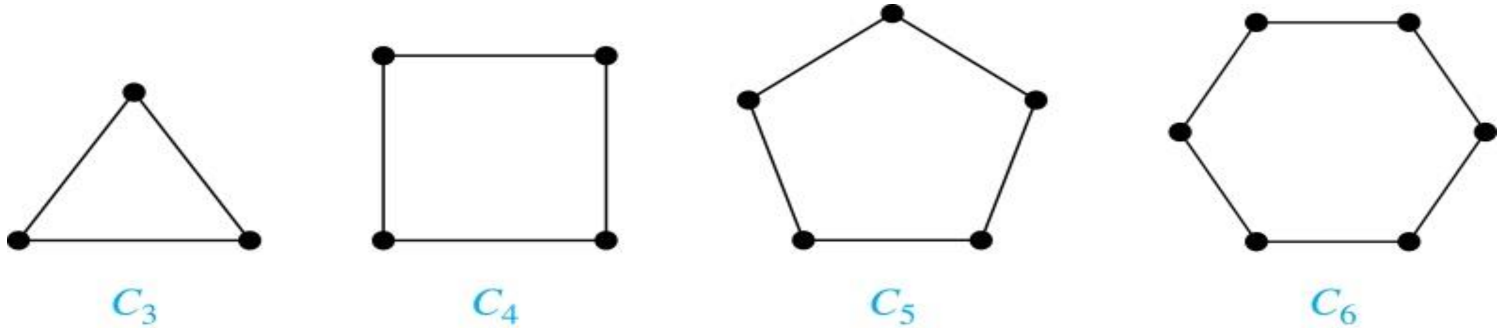


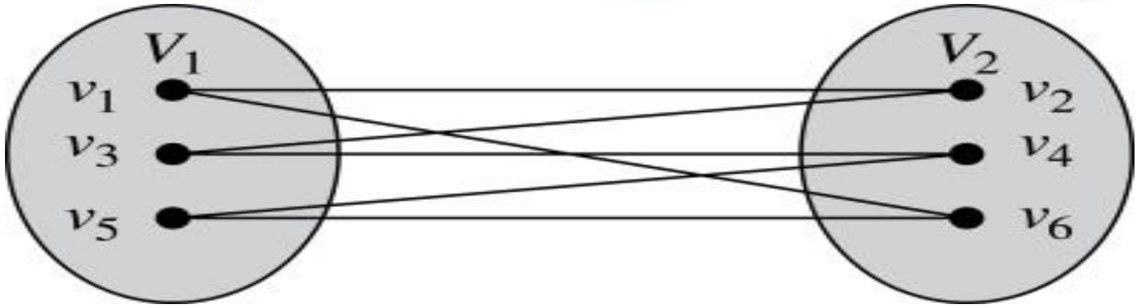
Bipartite Graphs (*continued*)

Example: Show that C_6 is bipartite.

Solution: We can partition the vertex set into

$V_1 = \{v_1, v_3, v_5\}$ and $V_2 = \{v_2, v_4, v_6\}$ so that every edge of C_6 connects a vertex in V_1 and V_2 .

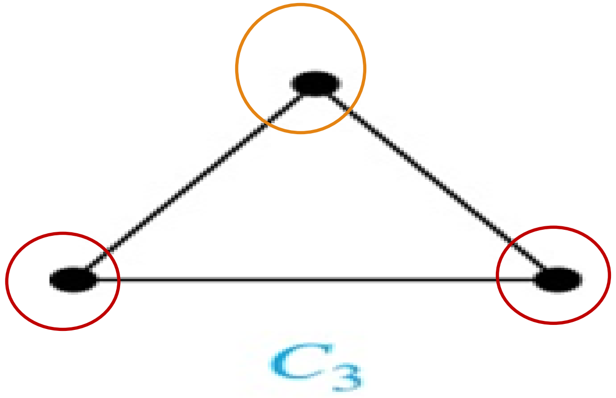




Bipartite Graphs (*continued*)

Example: Show that C_3 is not bipartite.

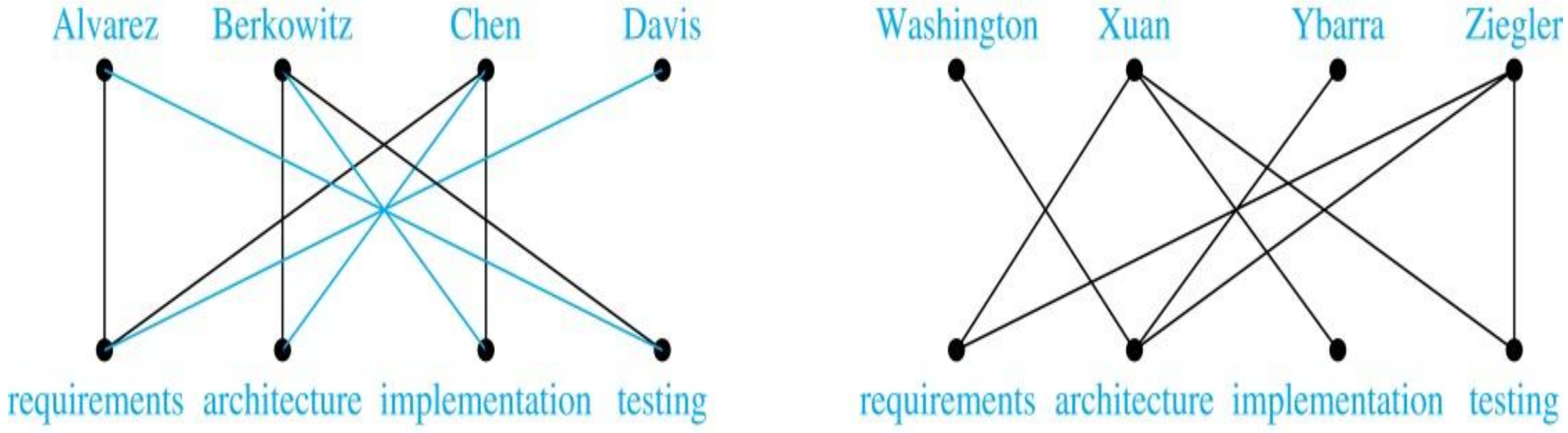
Solution: If we divide the vertex set of C_3 into two nonempty sets, one of the two must contain two vertices. But in C_3 every vertex is connected to every other vertex. Therefore, the two vertices in the same partition are connected. Hence, C_3 is not bipartite.



Bipartite Graphs and Matchings

Bipartite graphs are used to model applications that involve matching the elements of one set to elements in another, for example:

Job assignments - vertices represent the jobs and the employees, edges link employees with those jobs they have been trained to do. A common goal is to match jobs to employees so that the most jobs are done.

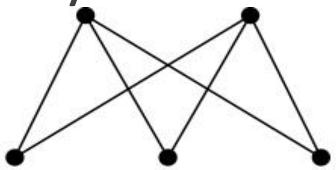


(a) (b)

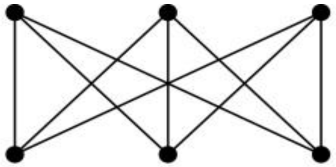
Complete Bipartite Graphs

Definition: A *complete bipartite graph* $K_{m,n}$ is a graph that has its vertex set partitioned into two subsets V_1 of size m and V_2 of size n such that there is an edge from every vertex in V_1 to every vertex in V_2.

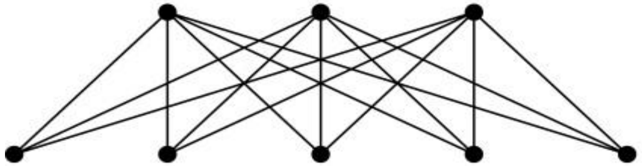
Example: We display four complete bipartite graphs here.



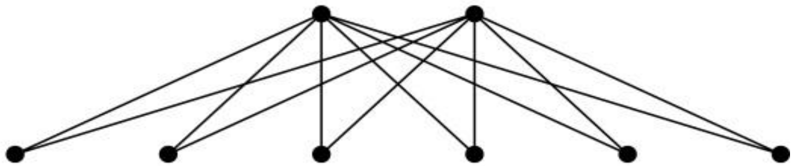
$K_{2,3}$



$K_{3,3}$



$K_{3,5}$

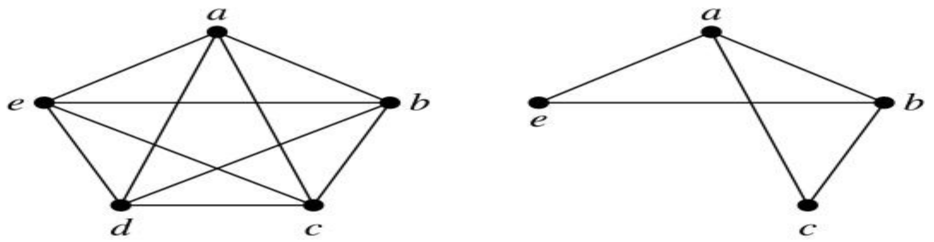


$K_{2,6}$

New Graphs from Old

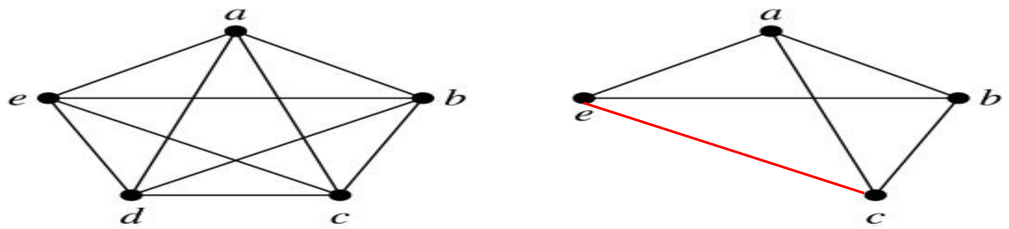
Definition: A *subgraph of a graph* $G = (V,E)$ is a graph (W,F), where $W \subset V$ and $F \subset E$. A subgraph H of G is a proper subgraph of G if $H \neq G$. A general subgraph can have less edges between the same vertices than the original one.

Example: Here we show K_5 and one of its subgraphs.



Definition: Let $G = (V, E)$ be a simple graph. The *subgraph induced* by a subset W of the vertex set V is the graph (W,F), where the edge set F contains an edge in E if and only if both endpoints are in W.

Example: Here we show K_5 and the subgraph induced by $W = \{a,b,c,e\}$.



New Graphs from Old (*continued*)

Definition: The *union* of two simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is the simple graph with vertex set $V_1 \cup V_2$ and edge set $E_1 \cup E_2$. The union of G_1 and G_2 is denoted by $G_1 \cup G_2$.

Example: