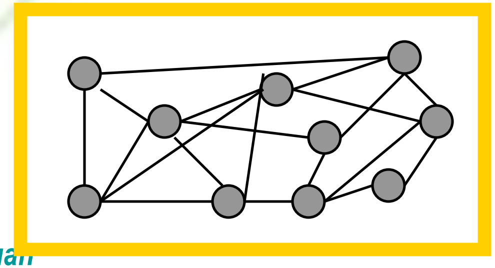
What are Graphs?

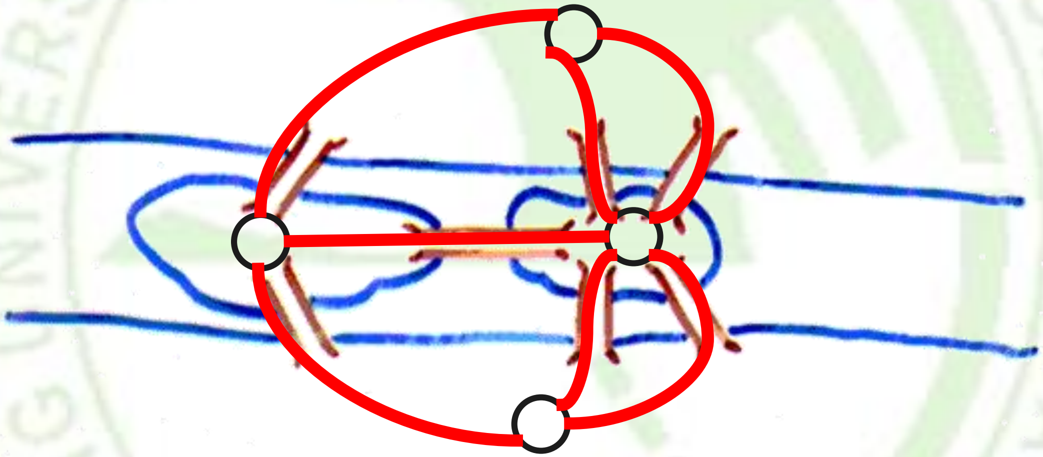
- General meaning in everyday math:
  A plot or chart of numerical data using a coordinate system.
- Technical meaning in discrete mathematics:
  A particular class of discrete structures (to be defined) that is useful for representing relations and has a convenient webby-looking graphical representation.



Königsberg Bridge problem哥尼斯堡七桥问题

- The residents of ***Königsberg***, ***Germany***, wondered if it was possible to take a walking tour of the town that crossed each of the seven bridges over the Pregel River exactly once.
  - Leonard Euler 1736 (father of graph theory) 欧拉

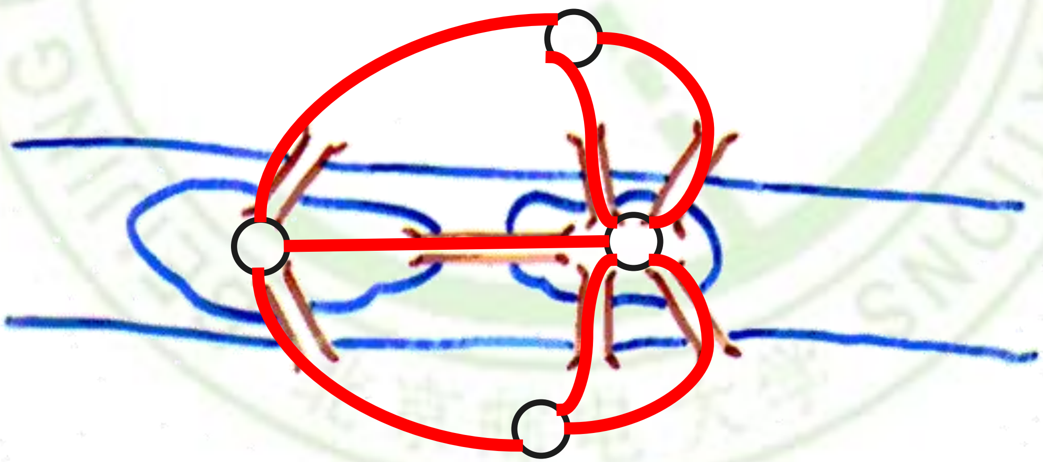
Königsberg Bridge problem



- Picture only what is essential to the problem.

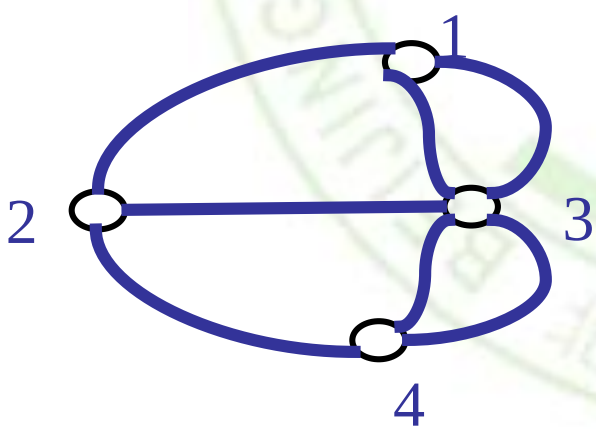
Königsberg Bridge problem

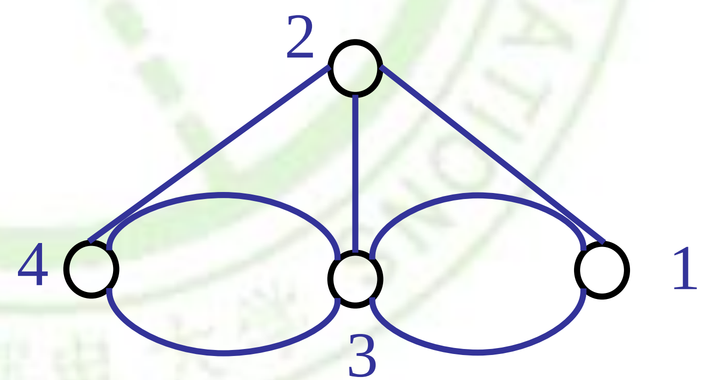
- Is it possible to start at some node and take a walk that uses each edge exactly once, and ends at the starting node?



Königsberg Bridge problem

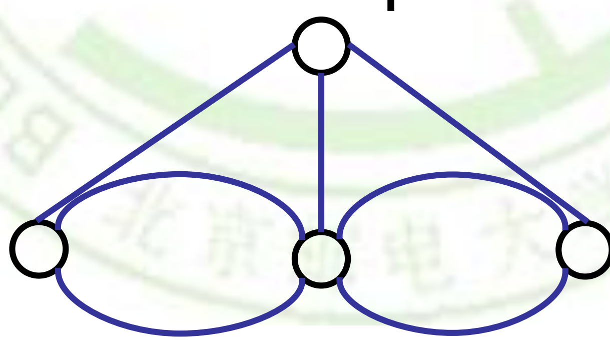
- You can redraw the original picture as long as for every edge between nodes *i* and *j* in the original you put an edge between nodes *i* and *j* in the redrawn version (and you put no other edges in the redrawn version)





Can you see why?

- Euler
  - Has no tour that uses each edge exactly once. 不存在一条简单回路
  - (Even if we allow the walk to start and finish in different places.)不存在一条简单通路



Applications of Graphs

- Potentially anything (graphs can represent relations, relations can describe the extension of any predicate).
- Apps in networking, scheduling, flow optimization, circuit design, path planning.
- More apps: Geneology analysis, computer game-playing, program compilation, object-oriented design, …

Types of Graphs

- Simple graph
- Multigraph
- Pseudograph
- Directed graph
- Directed multigraph

P642

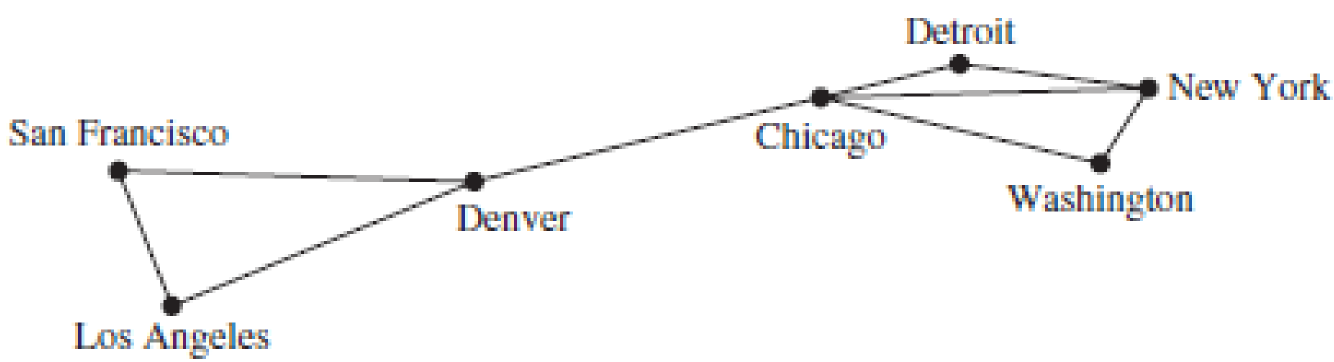
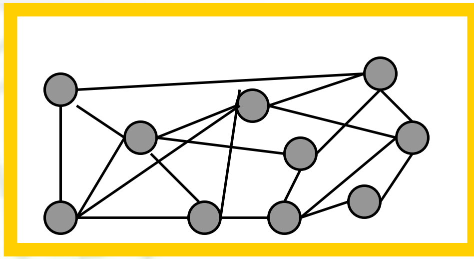


FIGURE 1 A Computer Network.

Simple Graphs简单图

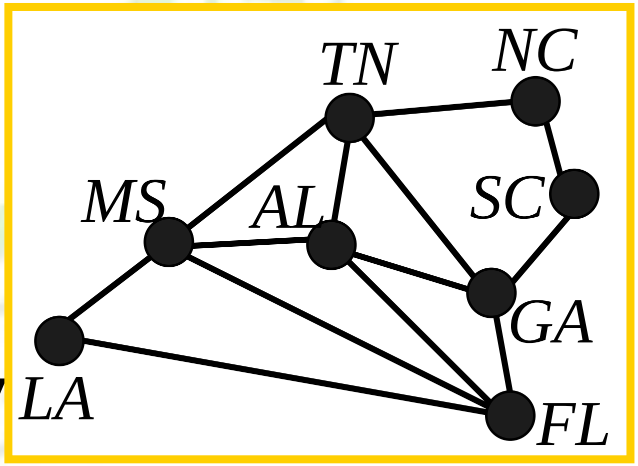
- Correspond to symmetric, irreflexive binary relations *R*.
- A *simple graph* $G=(V,E)$ consists of:
  - *V* ,a nonempty set of *vertices* or *nodes*
  - *E* ,a set of *edges* / *arcs* / *links*: unordered pairs of distinct elements $u,v \in V$, such that uRv.



Visual Representation of a Simple Graph

Example of a *Simple* Graph

- Let V be the set of states in the far-southeastern U.S.:
  - *I.e.*, V={FL, GA, AL, MS, LA, SC, TN, NC}
- Let E={{u,v}|u adjoins v}
  ={{FL,GA},{FL,AL},{FL,MS},
  {FL,LA},{GA,AL},{AL,MS},
  {MS,LA},{GA,SC},{GA,TN},
  {SC,NC},{NC,TN},{MS,TN},
  {MS,AL}}



P642

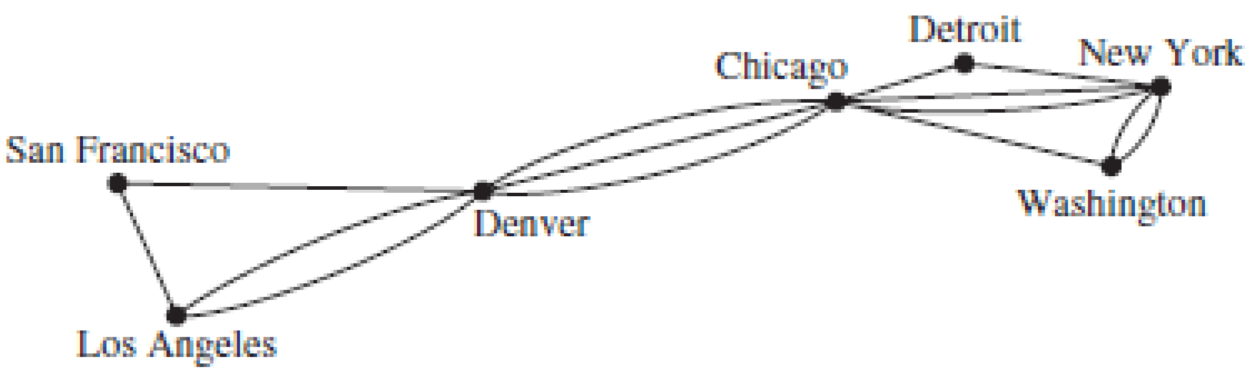
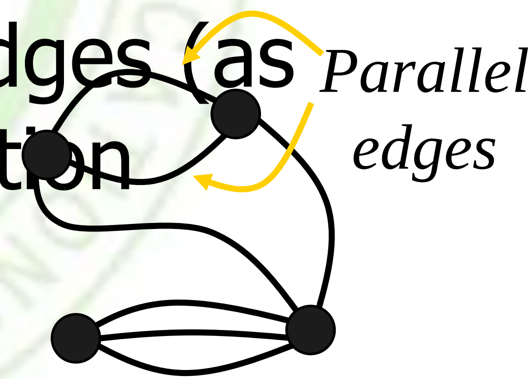


FIGURE 2 **A Computer Network with Multiple Links between Data Centers.**

Multigraphs 重图

- Like simple graphs, but there may be *more than one* edge connecting two given nodes.
- A *multigraph* $G=(V, E, f)$ consists of a set V of vertices, a set E of edges (as primitive objects), and a function $f: E \rightarrow \{\{u,v\} \mid u,v \in V \wedge u \neq v\}$.
- *E.g.*, nodes are cities, edges are segments of major highways.



P642

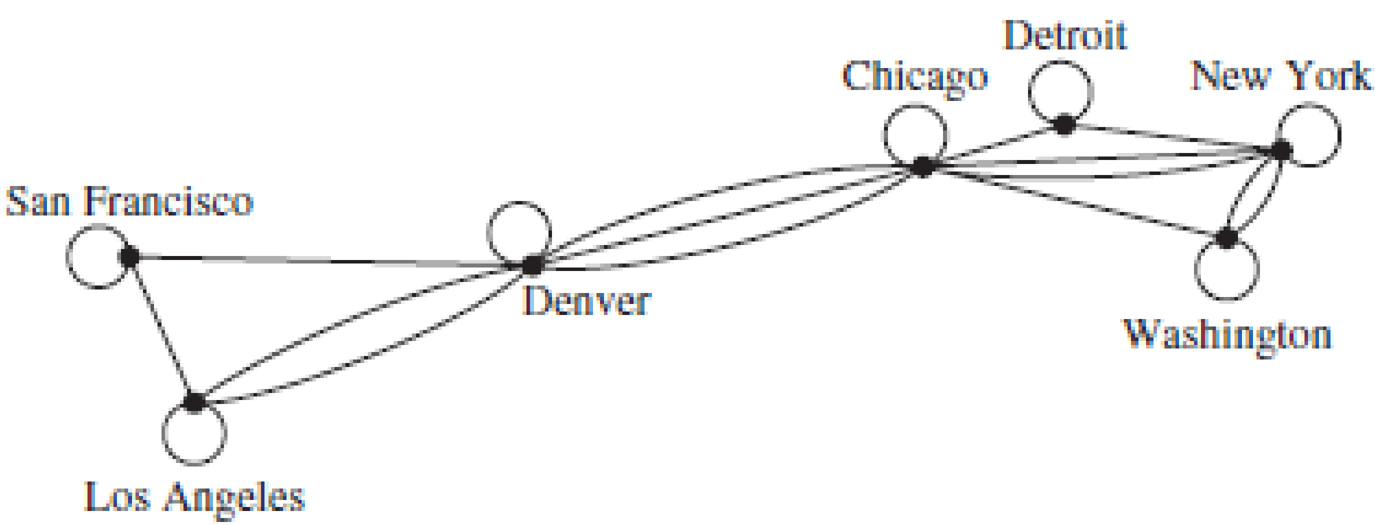
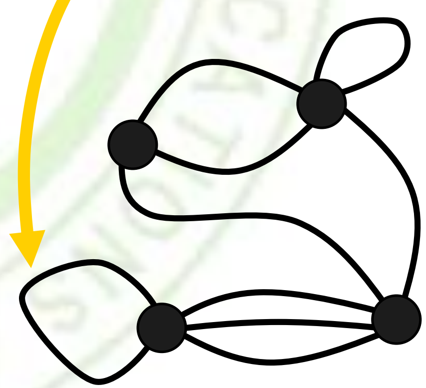


FIGURE 3 **A Computer Network with Diagnostic Links.**

Pseudographs伪图

- Like a multigraph, but edges connecting a node to itself are allowed. (*R* may be reflexive.)
- A *pseudograph* $G=(V, E, f)$ where $f:E\rightarrow\{\{u,v\}|u,v\in V\}$. Edge $e\in E$ is a *loop* if $f(e)=\{u,u\}=\{u\}$.
- *E.g.*, nodes are campsites in a state park, edges are hiking trails through the woods.



P643

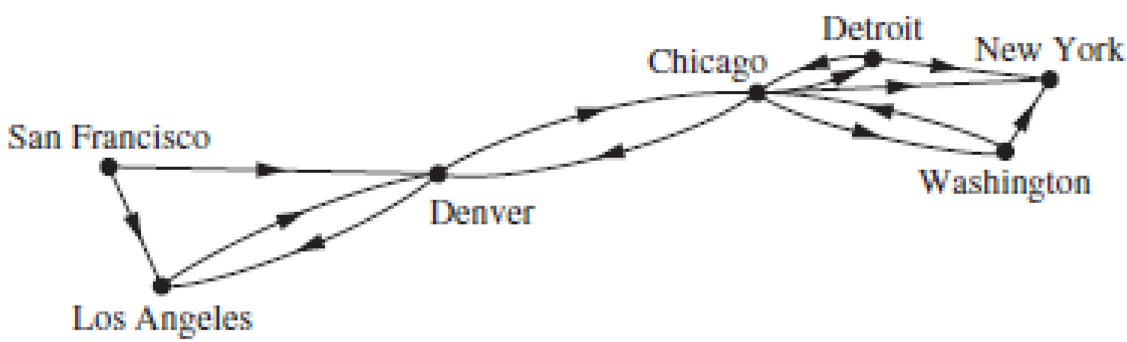
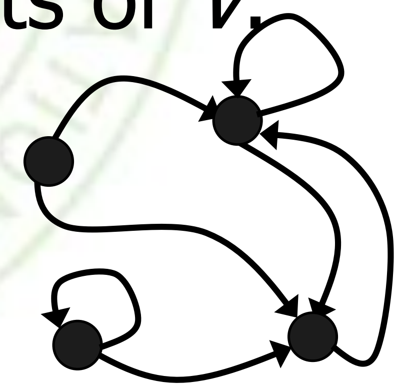


FIGURE 4 A Communications Network with One-Way Communications Links.

Directed Graphs有向图

- Correspond to arbitrary binary relations *R*, which need not be symmetric.
- A *directed graph* (*V*,*E*) consists of a set of vertices *V* and a set of edges *E* that are ordered pairs of elements of *V*.
- *E.g.*: *V* = set of People,
  E={(*x*,*y*) | *x* loves *y*}



P643

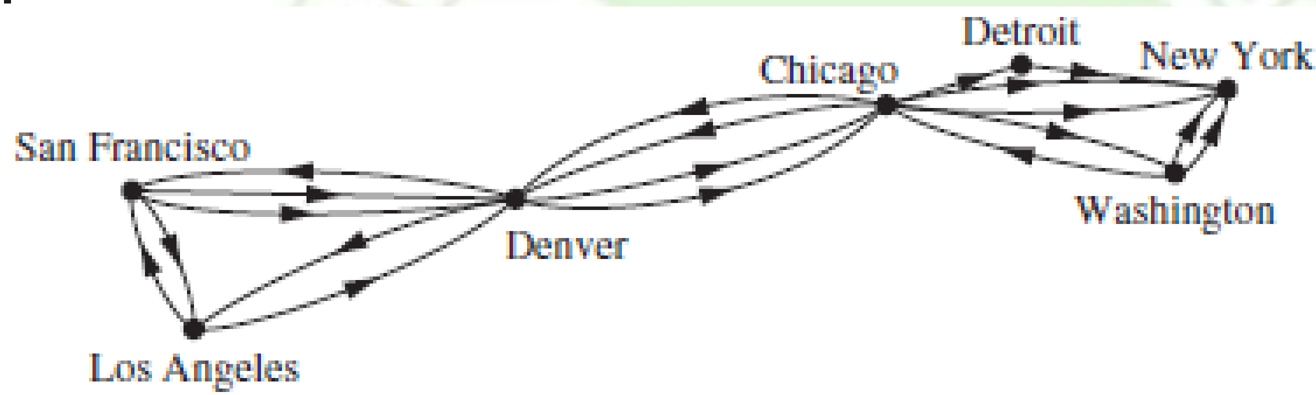
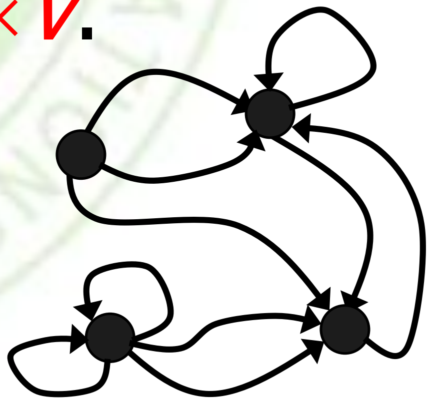


FIGURE 5 **A Computer Network with Multiple One-Way Links.**

Directed Multigraphs

- Like directed graphs, but there may be more than one arc from a node to another.
- A *directed multigraph* $G=(V, E, f)$ consists of a set V of vertices, a set E of edges, and a function $f:E\rightarrow V\times V$.
- *E.g.*, V=web pages, E=hyperlinks. *The WWW is a directed multigraph...*



Types of Graphs: Summary

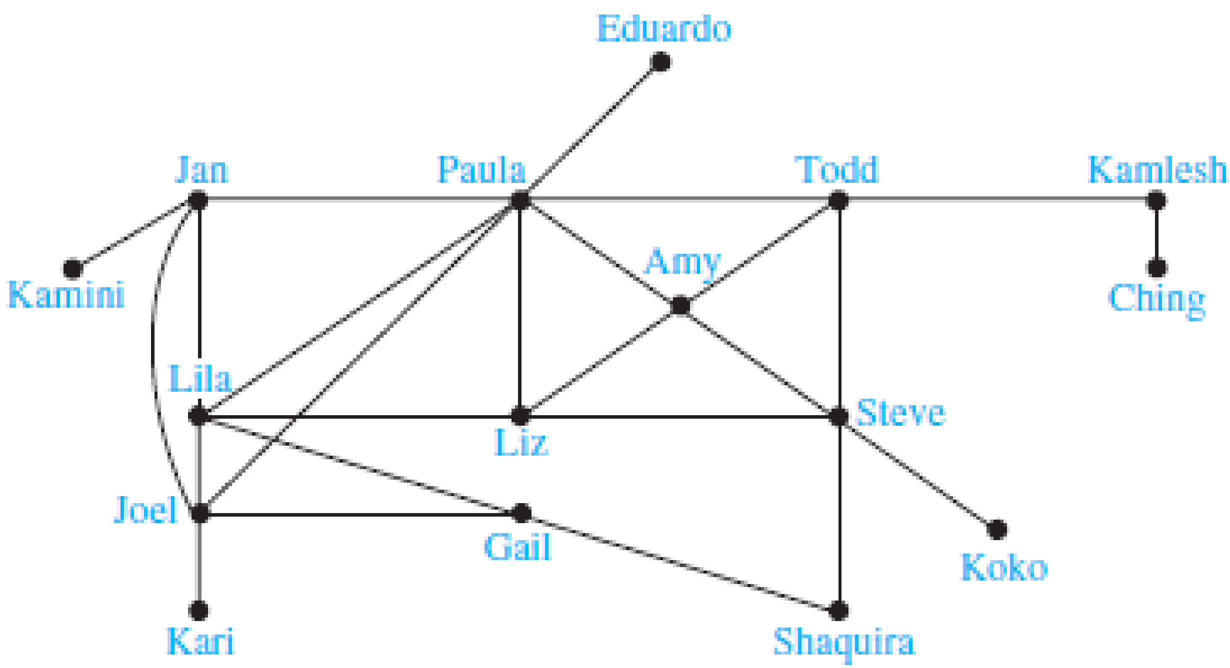
- Summary of the book's definitions.
- Keep in mind this terminology is not fully standardized across different authors...

Term	Edge type	Multiple edges ok?	Self-loops ok?
Simple graph	Undir.	No	No
Multigraph	Undir.	Yes	No
Pseudograph	Undir.	Yes	Yes
Directed graph	Directed	No	Yes
Directed multigraph	Directed	Yes	Yes

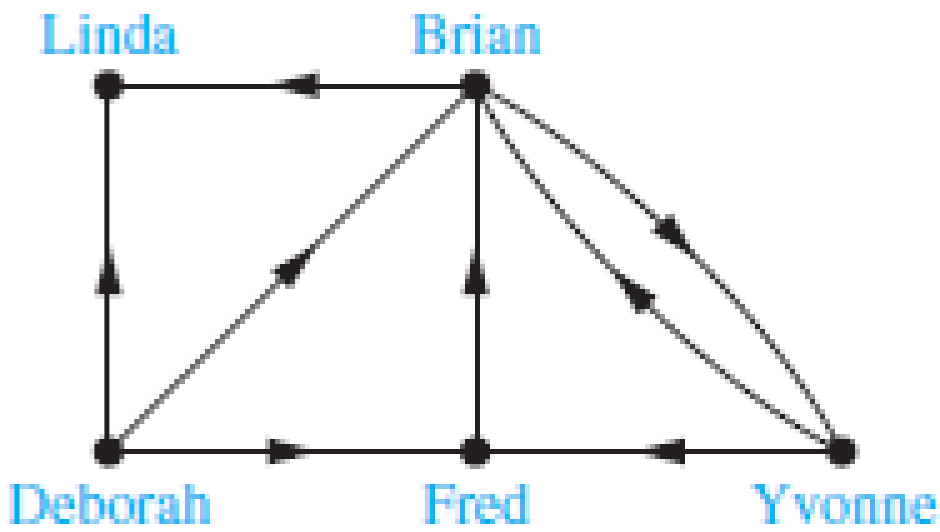
Graph models

- Social Networks
- Information Networks
- Software Design Applications
- Transportation Networks
- Biological Networks
- Tournaments

Acquaintanceship Graph
相识图



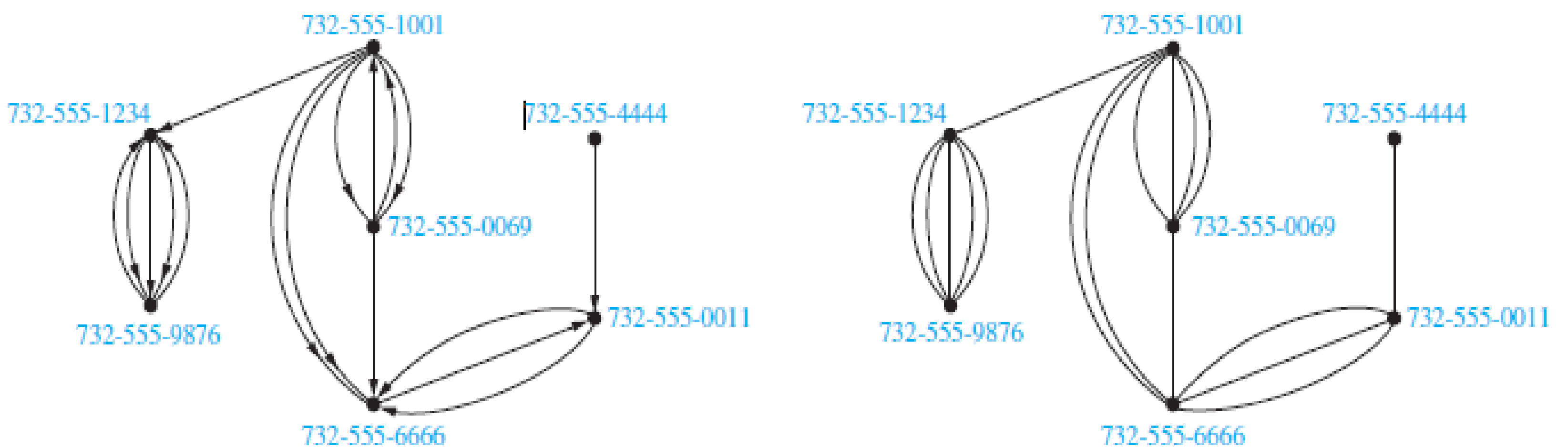
Influence Graphs影响力图



Collaboration Graphs合作图

- A collaboration graph is used to model social networks where two people are related by working together in a particular way.

Call graphs呼叫图



(a) (b)

The Web graph

- The WorldWideWeb can be modeled as a directed graph where each Web page is represented by a vertex and where an edge starts at the Web page *a* and ends at the Web page *b* if there is a link on *a* pointing to *b*.

MODULE DEPENDENCY GRAPH
模块依赖关系图

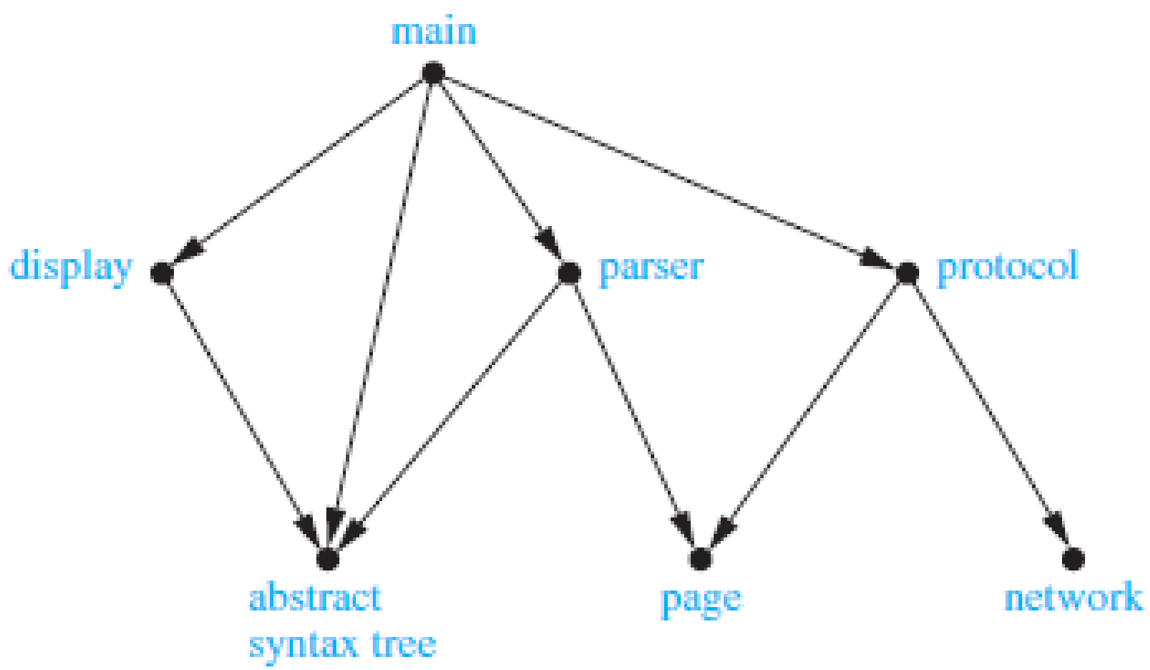
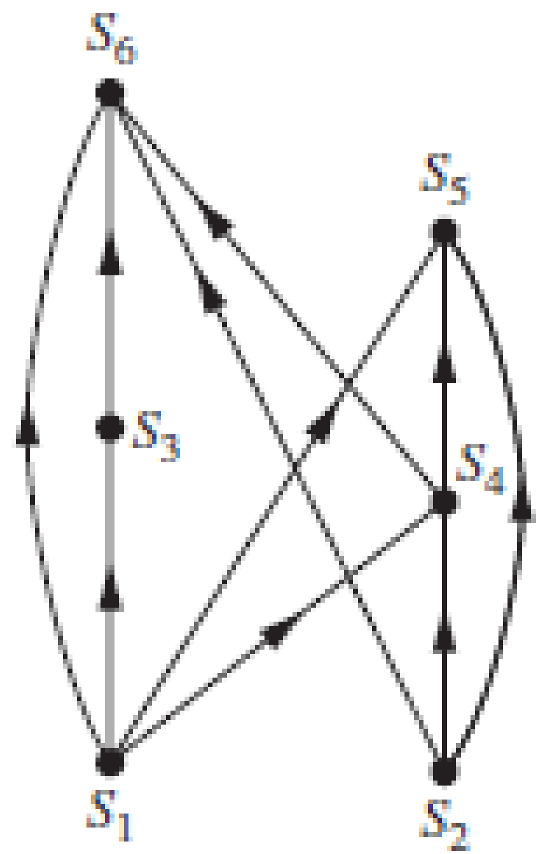


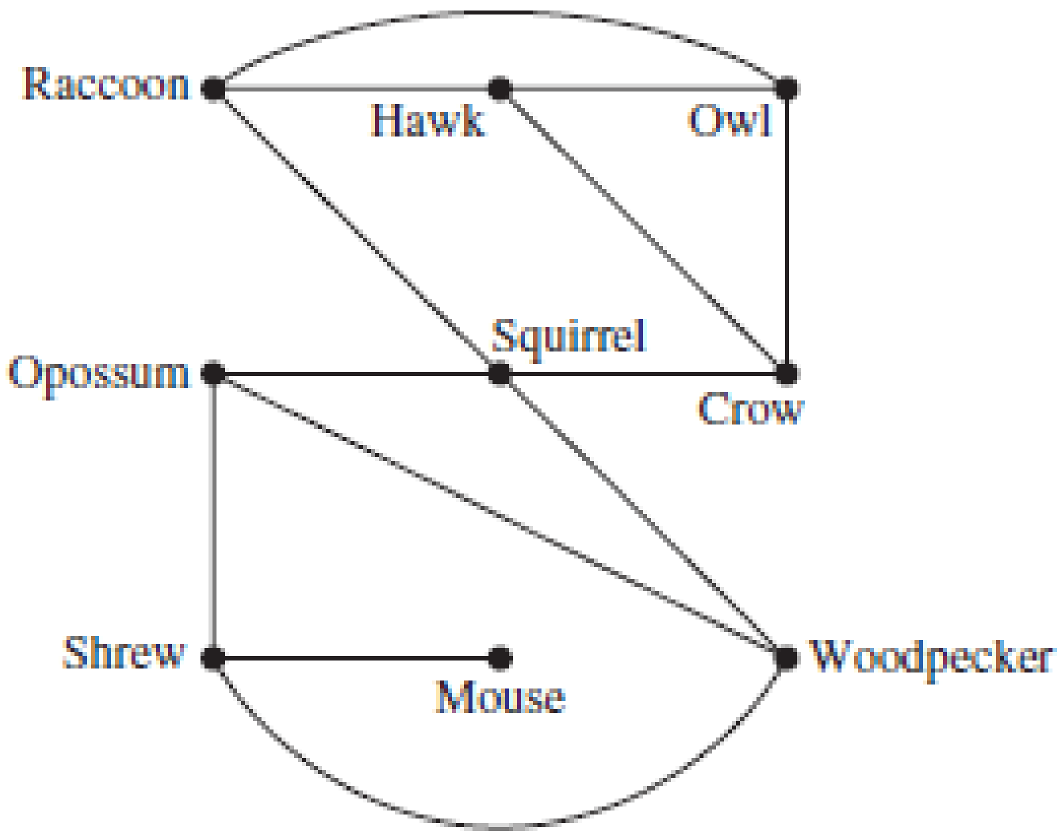
FIGURE 9 **A Module Dependency Graph.**

Precedence Graphs and Concurrent Processing并发处理中的优先性图

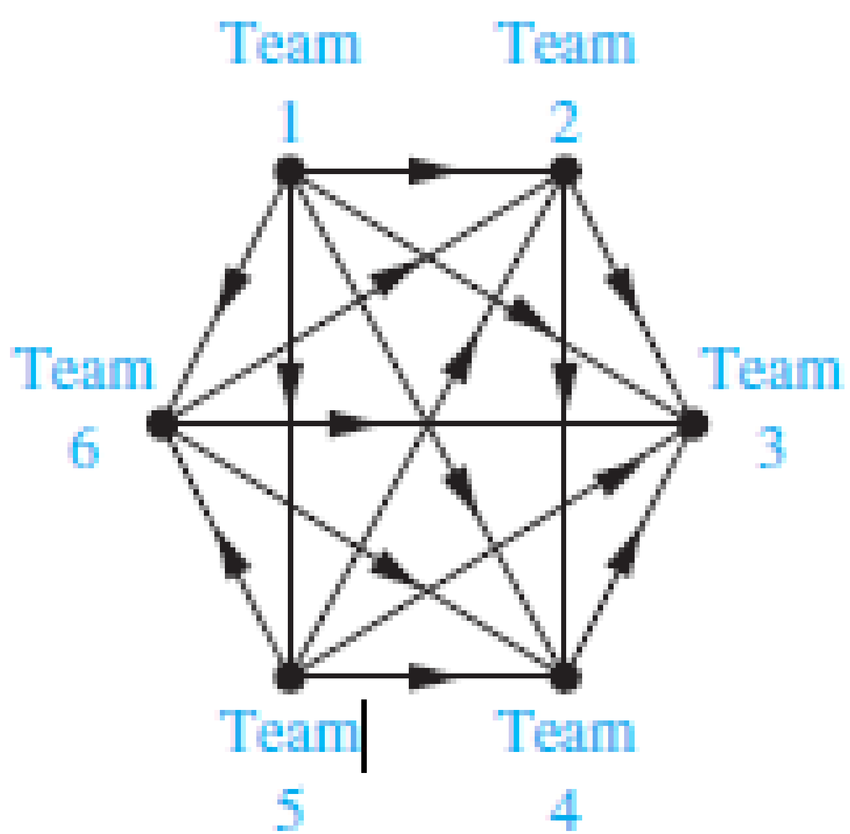
S_1	$a := 0$
S_2	$b := 1$
S_3	$c := a + 1$
S_4	$d := b + a$
S_5	$e := d + 1$
S_6	$e := c + d$



Niche Overlap Graphs in Ecology



Round-Robin Tournaments 循环锦标赛



Homework

- § 10.1
  - 22,34

§ 10.2 Graph Terminology

You need to learn the following terms:

- *Adjacent, connects, endpoints, degree, initial, terminal, in-degree, out-degree, complete, cycles, wheels, n-cubes, bipartite, subgraph, union.*

Adjacency邻接

Let G be an undirected graph with edge set E. Let $e \in E$ be (or map to) the pair $\{u,v\}$. Then we say:

- *u*, *v* are *adjacent* / *neighbors* / *connected*.
- Edge *e* is *incident with* vertices *u* and *v*.
- Edge *e connects u* and *v*.
- Vertices *u* and *v* are *endpoints* of edge *e*.

Neighborhood邻域

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

Degree of a Vertex顶点的度

- Let *G* be an undirected graph, $v \in V$ a vertex.
- The *degree* of *v*, deg(*v*), is its number of incident edges. (Except that any self-loops are counted twice.)
- A vertex with degree 0 is called *isolated*. 孤立点
- A vertex of degree 1 is called *pendant*. 悬挂点

Example 1

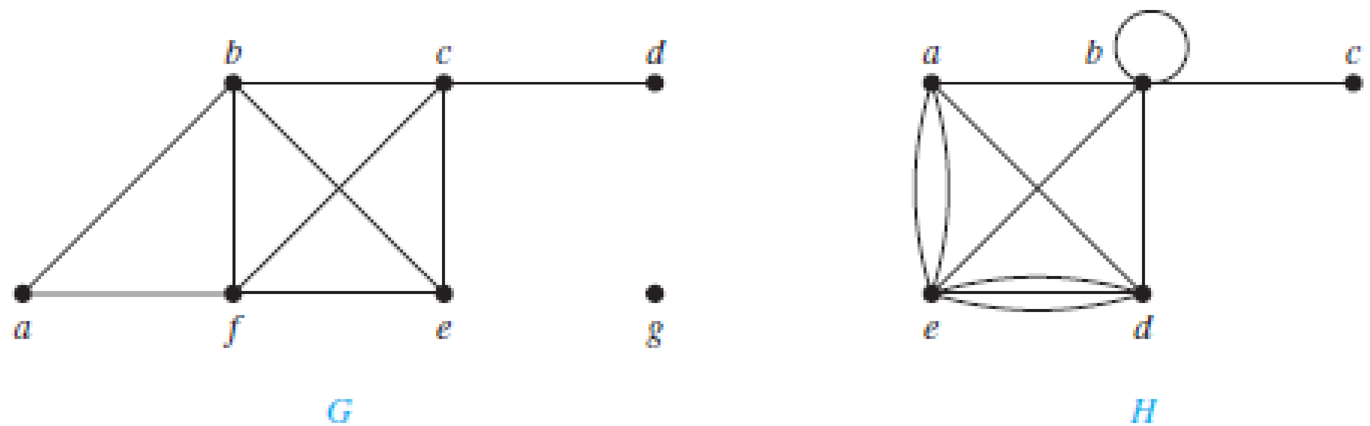


FIGURE 1 The Undirected Graphs G and H.

Handshaking Theorem握手定理

- Let *G* be an undirected (simple, multi-, or pseudo-) graph with vertex set *V* and edge set *E*. Then

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

- **Corollary:** Any undirected graph has an even number of vertices of odd degree.奇数度的结点数一定是偶数。

Directed Adjacency有向邻接

- Let *G* be a directed (possibly multi-) graph, and let *e* be an edge of *G* that is (or maps to) (*u*,*v*). Then we say:
  - *u* is *adjacent to v*, *v* is *adjacent from u*
  - *e comes from* u, e *goes to* v.
  - *e connects u to v*, *e goes from u to v*
  - the *initial vertex* of *e* is *u 边e的起点*
  - the *terminal vertex* of *e* is *v 边e的终点*

Directed Degree

- Let G be a directed graph, v a vertex of G.
  - The *in-degree* of v, $\deg^{-}(v)$, is the number of edges going to v. 入度
  - The *out-degree* of v, $\deg^{+}(v)$, is the number of edges coming from v. 出度
  - The *degree* of v, $\deg(v) :\equiv \deg^{-}(v) + \deg^{+}(v)$, is the sum of v's in-degree and out-degree.

Example 4

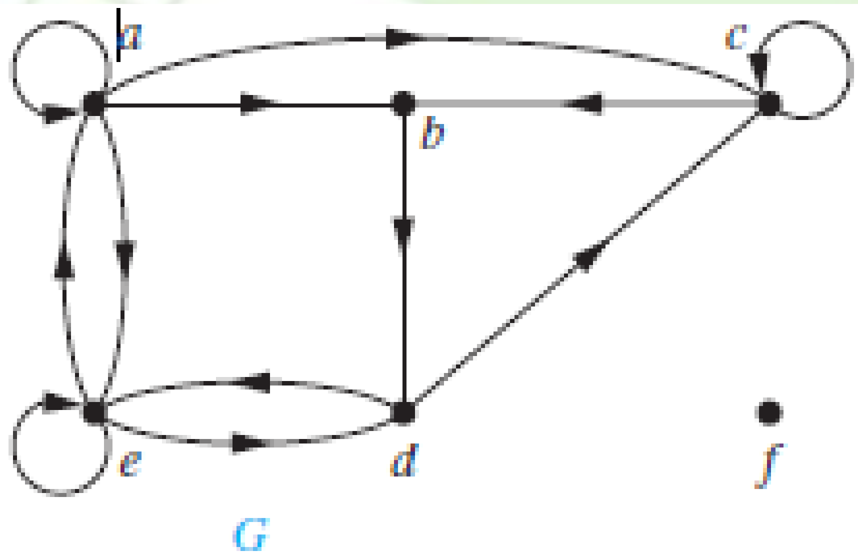


FIGURE 2 The Directed Graph G.

Directed Handshaking Theorem

- Let G be a directed (possibly multi-) graph with vertex set V and edge set E.

$$\sum_{v\in V}\deg^{-}(v)=\sum_{v\in V}\deg^{+}(v)=\frac{1}{2}\sum_{v\in V}\deg(v)=|E|$$

- Note that the degree of a node is unchanged by whether we consider its edges to be directed or undirected.

Underlying undirected graph基本无向图

- The undirected graph that results from ignoring directions of edges is called the underlying undirected graph.

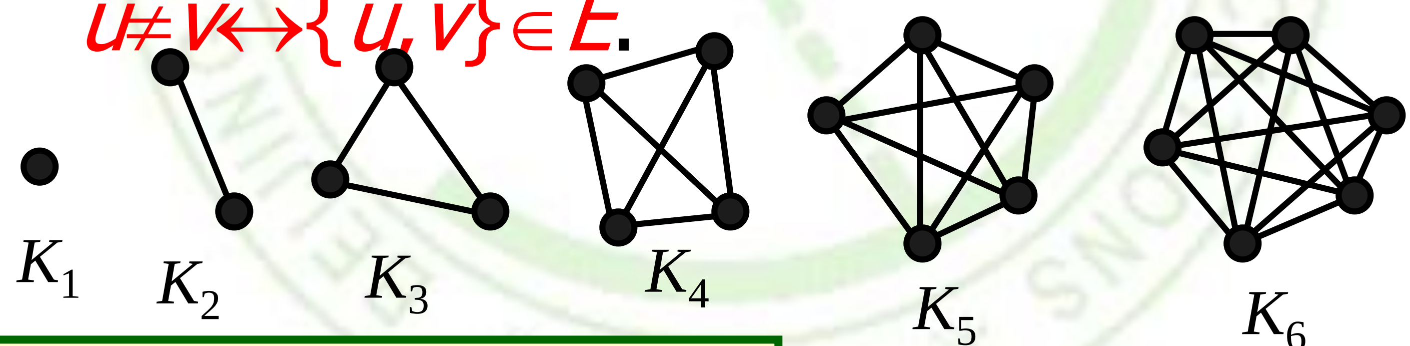
Special Graph Structures

Special cases of undirected graph structures:

- Complete graphs K_n
- Cycles C_n
- Wheels W_n
- n-Cubes Q_n
- Bipartite graphs
- Complete bipartite graphs $K_{m,n}$

Complete Graphs完全图/连通图

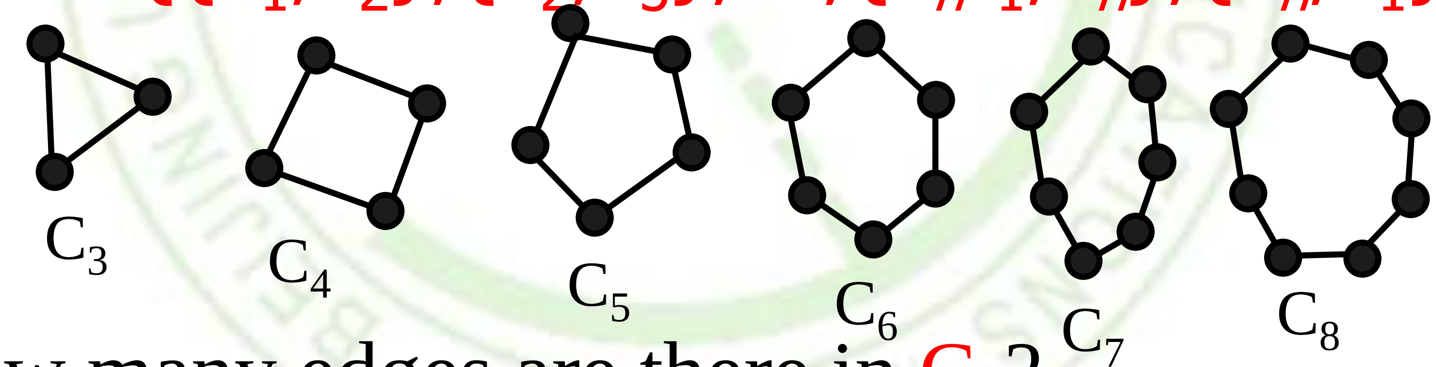
- For any $n \in \mathbf{N}$, a *complete graph* on n vertices, K_n, is a simple graph with n nodes in which every node is adjacent to every other node: $\forall u,v \in V$: $u \neq v \leftrightarrow \{u,v\} \in E$.



Note that K_n *has* $\sum_{i=1}^{n-1} i = \frac{n(n-1)}{2}$ *edges.*

Cycles环图

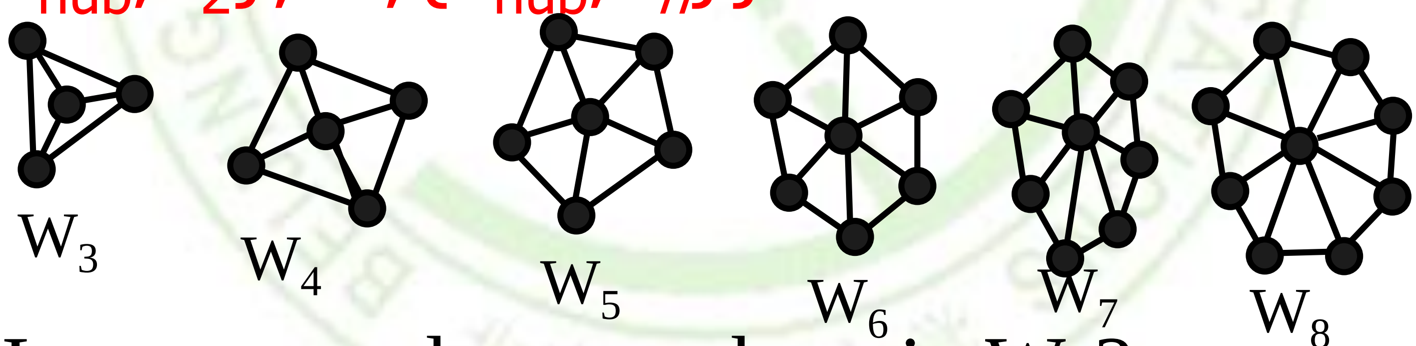
- For any $n \geq 3$, a *cycle* on n vertices, C_n, is a simple graph where $V=\{v_1, v_2, \ldots, v_n\}$ and
$E=\{\{v_1,v_2\},\{v_2,v_3\},\ldots,\{v_{n-1},v_n\},\{v_n,v_1\}\}$.



How many edges are there in C_n?

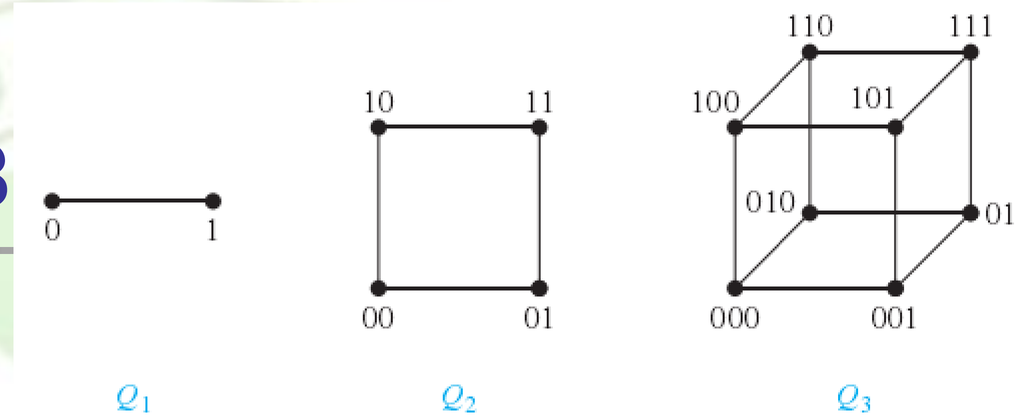
Wheels轮图

- For any $n \geq 3$, a *wheel* W_n, is a simple graph obtained by taking the cycle C_n and adding one extra vertex v_{hub} and n extra edges $\{\{v_{hub}, v_1\}, \{v_{hub}, v_2\}, \ldots, \{v_{hub}, v_n\}\}$.



How many edges are there in W_n?

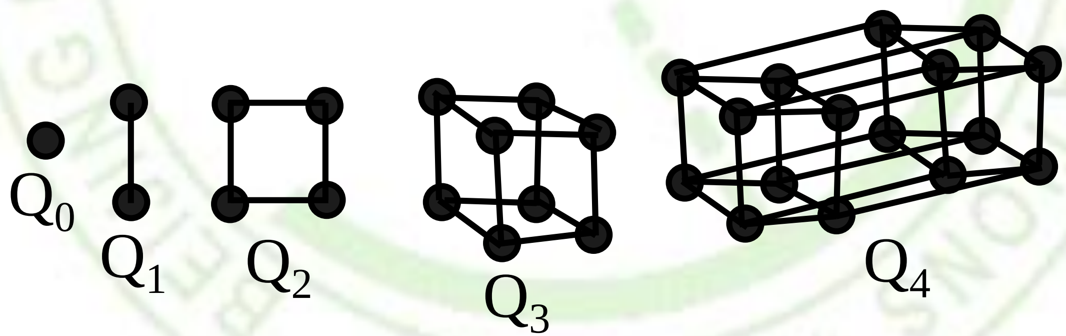
HYPERCUB



- An n-dimensional hypercube, or n-cube, denoted by Qn, is a 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.

n-cubes (hypercubes)超立方体

- For any $n \in \mathbf{N}$, the hypercube Q_n is a simple graph consisting of two copies of Q_{n-1} connected together at corresponding nodes. Q_0 has 1 node.



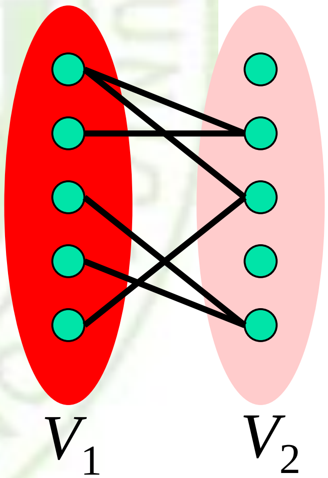
Number of vertices: 2^n. Number of edges: $n2^{n-1}$

n-cubes (hypercubes)

- For any $n \in \mathbf{N}$, the hypercube Q_n can be defined recursively as follows:
  - $Q_0 = \{\{v_0\}, \varnothing\}$ (one node and no edges)
  - For any $n \in \mathbf{N}$, if $Q_n = (V, E)$, where $V = \{v_1, \ldots, v_a\}$ and $E = \{e_1, \ldots, e_b\}$, then $Q_{n+1} = (V \cup \{v_1', \ldots, v_a'\}, E \cup \{e_1', \ldots, e_b'\} \cup \{\{v_1, v_1'\}, \{v_2, v_2'\}, \ldots, \{v_a, v_a'\}\})$ where $v_1', \ldots, v_a'$ are new vertices, and where if $e_i = \{v_j, v_k\}$ then $e_i' = \{v_j', v_k'\}$.

Bipartite Graphs二分图

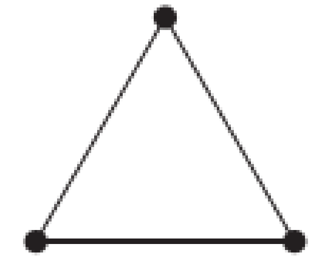
- **Def'n.:** A graph $G=(V,E)$ is *bipartite* (two-part) iff $V = V_1 \cup V_2$ where $V_1 \cap V_2 = \emptyset$ and $\forall e \in E: \exists v_1 \in V_1, v_2 \in V_2: e=\{v_1, v_2\}$.
- **In English:** The graph can be divided into two parts in such a way that all edges go between the two parts.



This definition can easily be adapted for the case of multigraphs and directed graphs as well.

Can represent with zero-one matrices.

Example 9



- Show K_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.

Example 10

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.

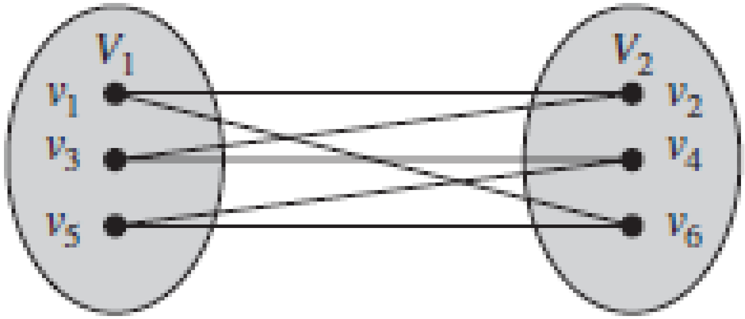


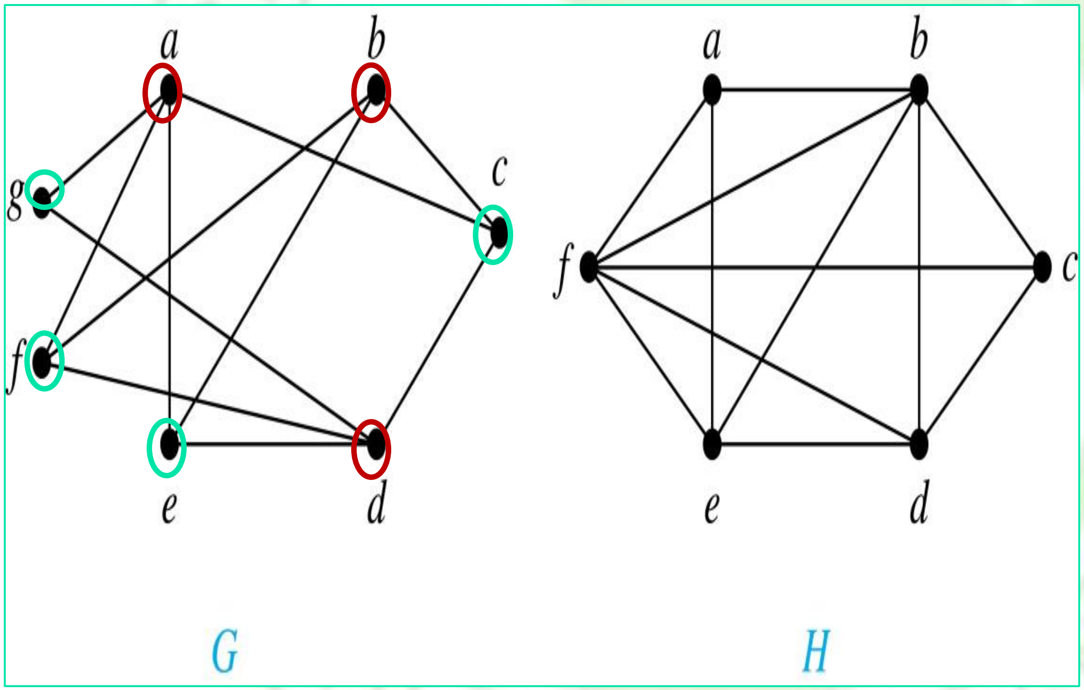
FIGURE 7 Showing That C_6 Is Bipartite.

Theorem二分图判断准则

- A simple graph is bipartite if and only if it is possible to assign one of two different colors to each vertex of the graph so that no two adjacent vertices are assigned the same color.
- 顶点着二色，没有两个邻接点着色相同。

Example 11

G is bipartite

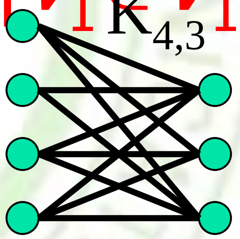


H is not bipartite since if we color *a* red, then the adjacent vertices *f* and *b* must both be blue.

Complete Bipartite Graphs完全二分图

- For $m,n \in \mathbf{N}$, the *complete bipartite graph* $K_{m,n}$ is a bipartite graph where $|V_1| = m$, $|V_2| = n$, and $E = \{\{v_1,v_2\} | v_1 \in V_1 \wedge v_2 \in V_2\}$.
  - That is, there are *m* nodes in the left part, *n* nodes in the right part, and every node in the left part is connected to every node in the right part.

$K_{4,3}$



$K_{m,n}$ has __ nodes and ______ edges.

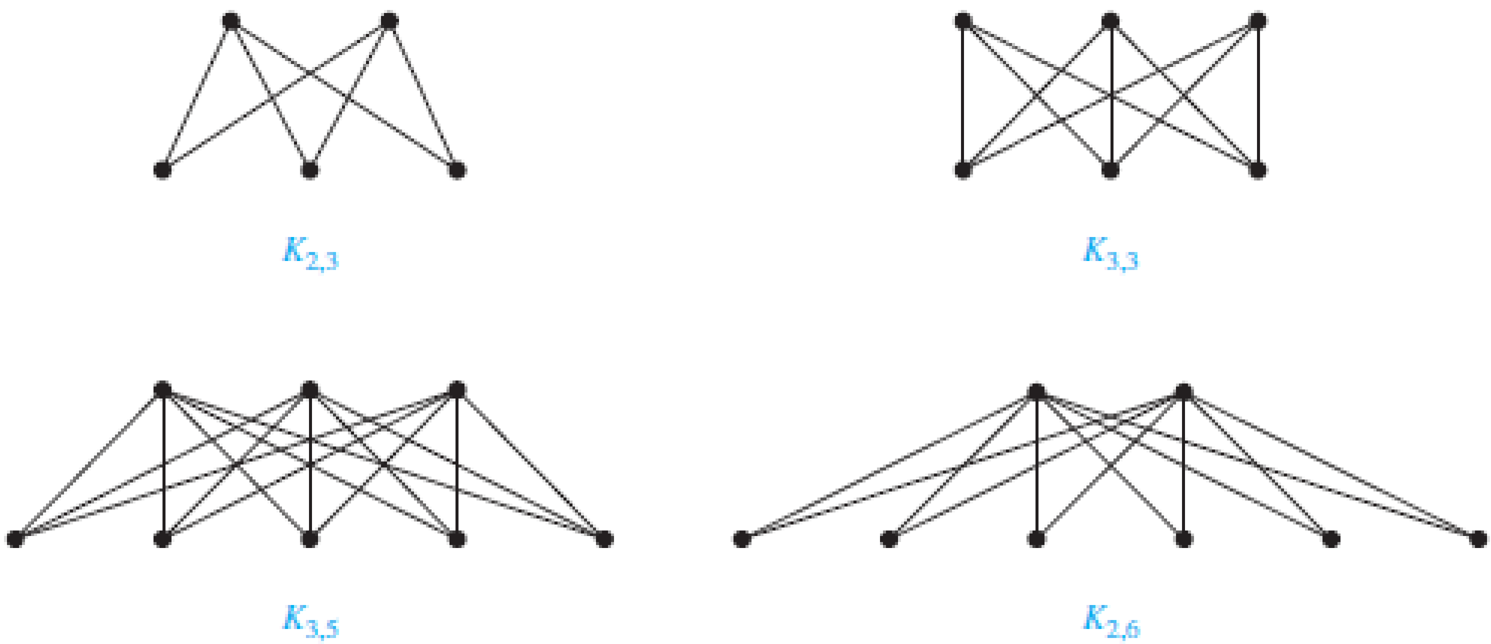


FIGURE 9 **Some Complete Bipartite Graphs.**

Bipartite Graphs and Matchings

- Job Assignments工作分配

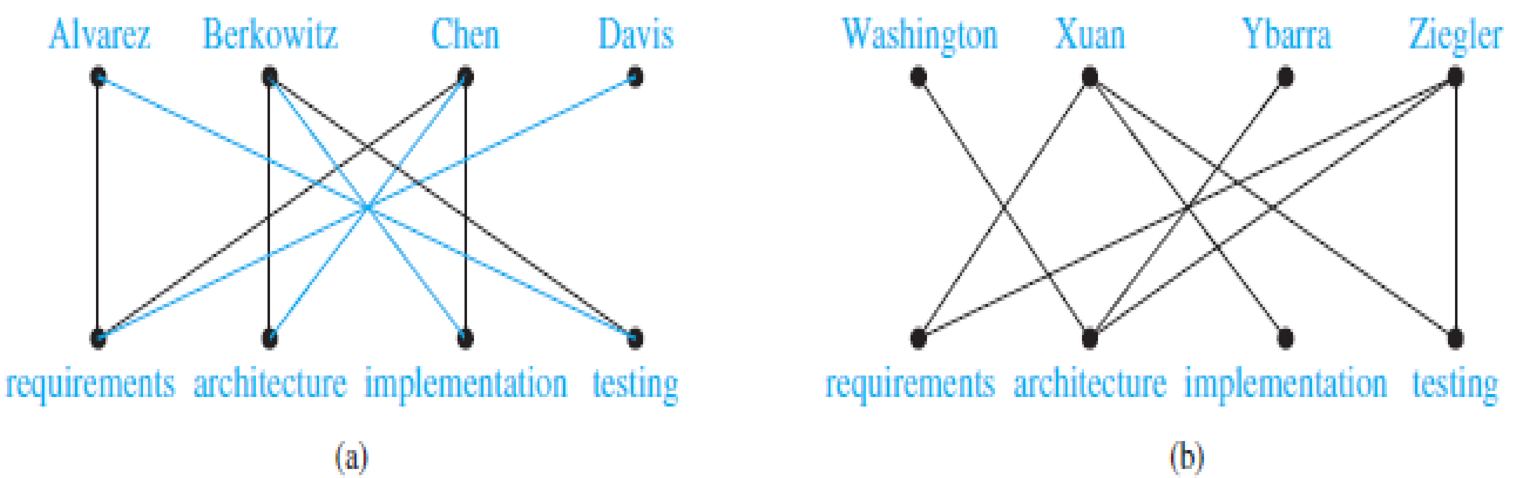


FIGURE 10 Modeling the Jobs for Which Employees Have Been Trained.

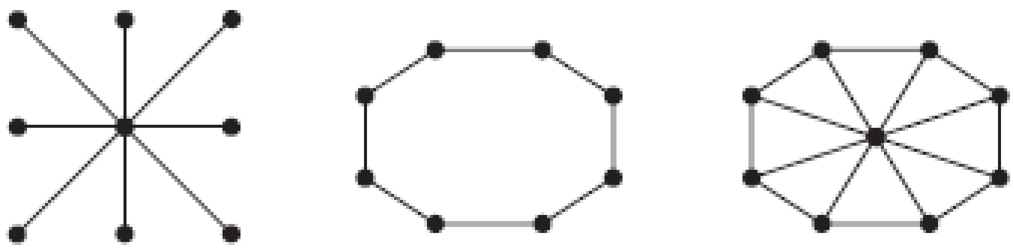
MATCHING匹配

- A matching *M* in a simple graph *G = (V ,E)* is a subset of the set *E* of edges of the graph such that no two edges are incident with the same vertex.
- A vertex that is the endpoint of an edge of a matching *M* is said to be **matched in *M;*** otherwise it is said to be **unmatched*.***
- A **maximum matching** is a matching with the largest number of edges. 最大匹配总是存在的
- A matching *M* in a bipartite graph *G = (V ,E)* with bipartition (V_1, V_2) *is a* **complete matching from V_1 to V_2** if every vertex in V_1 is the endpoint of an edge in the matching, or equivalently, if $|M| = |V_1|$. *完备匹配可能没有*

HALL'S MARRIAGE THEOREM霍尔婚姻定理

- The bipartite graph $G = (V, E)$ with bipartition (V_1, V_2) has a complete matching from V_1 to V_2 if and only if $|N(A)| \geq |A|$ for all subsets A of V_1.

Local Area Networks局域网



(a) (b) (c)

FIGURE 11 **Star, Ring, and Hybrid Topologies for Local Area Networks.**

Interconnection Networks for Parallel Computation

FIGURE 12 A Linear Array for Six Processors.

栅格网($N=M^2$,P(I,J)邻接P(I±1,J)和P(I,J±1))

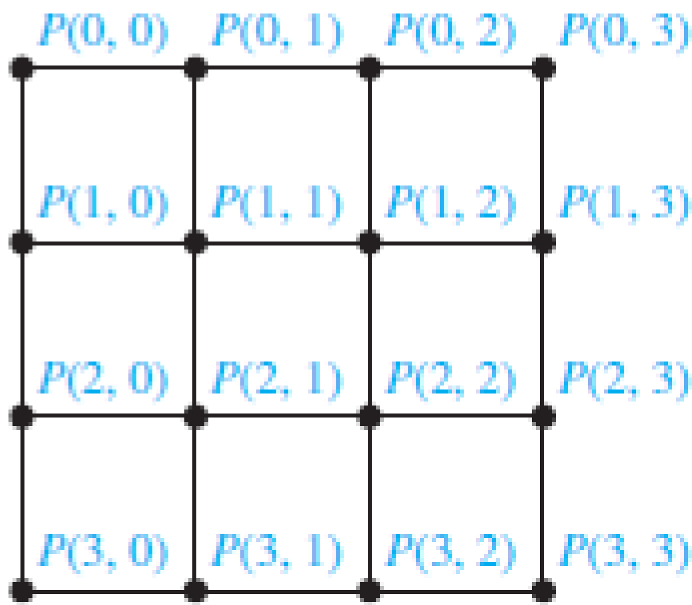


FIGURE 13 **A Mesh Network for 16 Processors.**

超立方体网 $N=2^M$

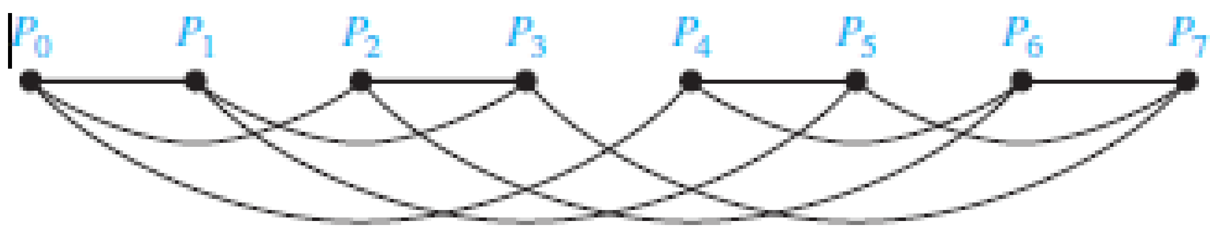
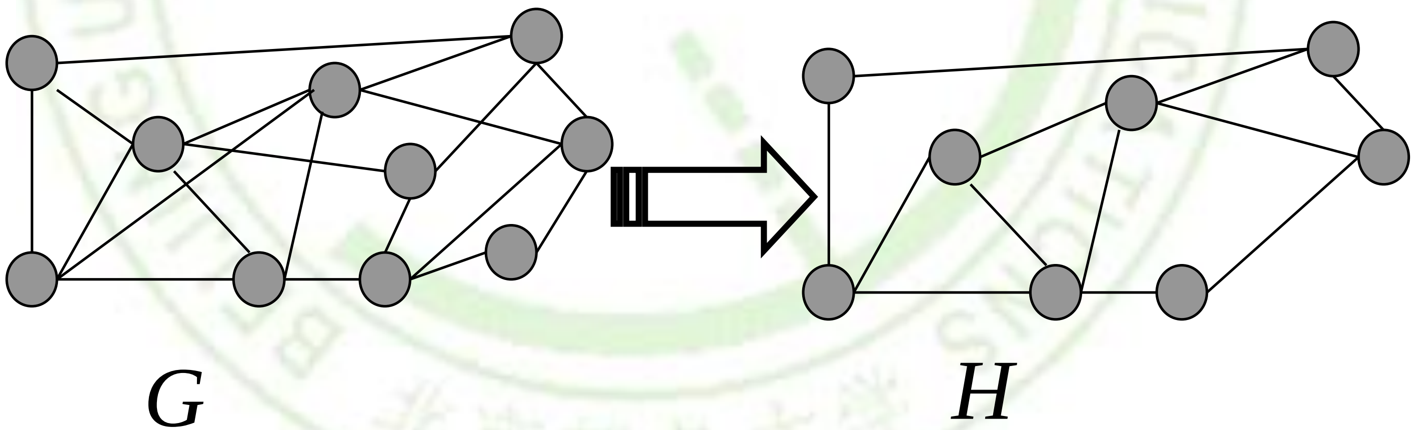


FIGURE 14 A Hypercube Network for Eight Processors.

Subgraphs子图

- 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 $H \neq G$. 真子图



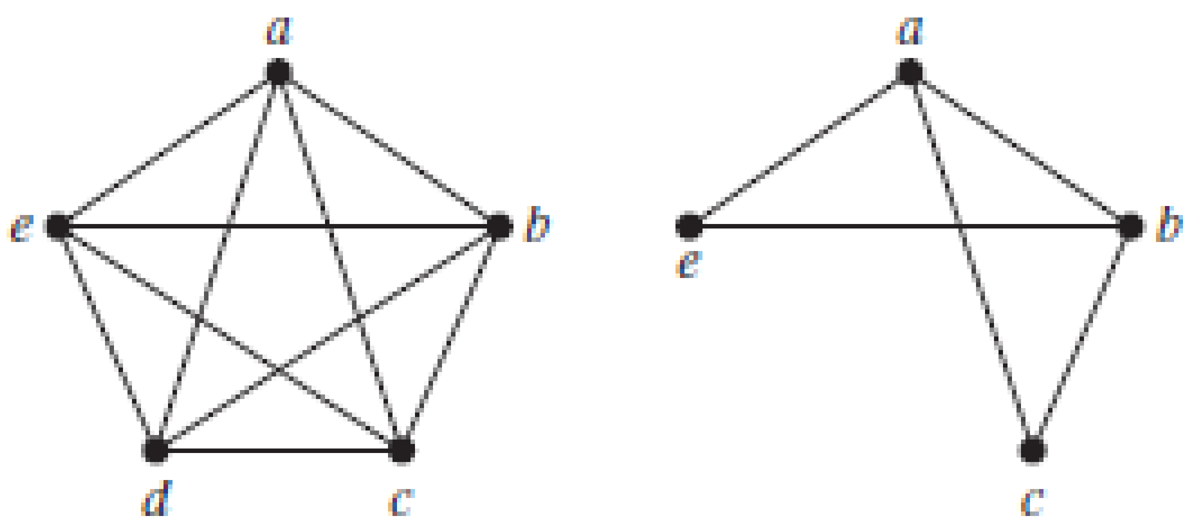
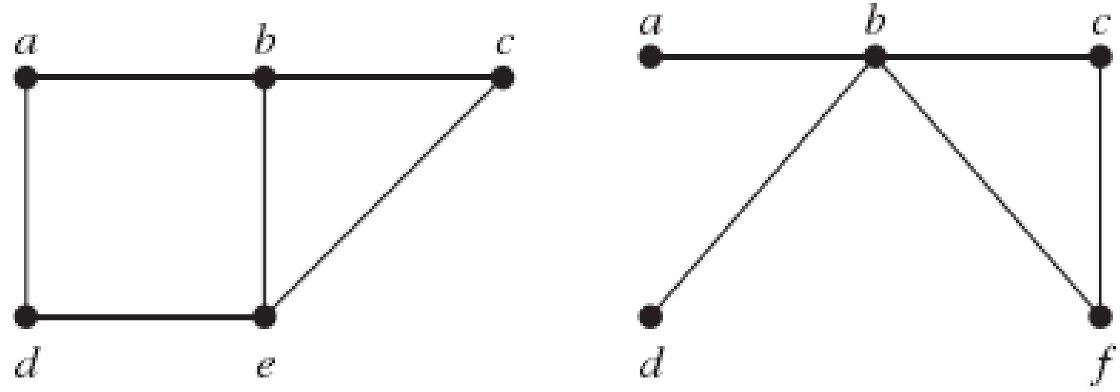


FIGURE 15 A Subgraph of K_5.

Subgraph-G_E

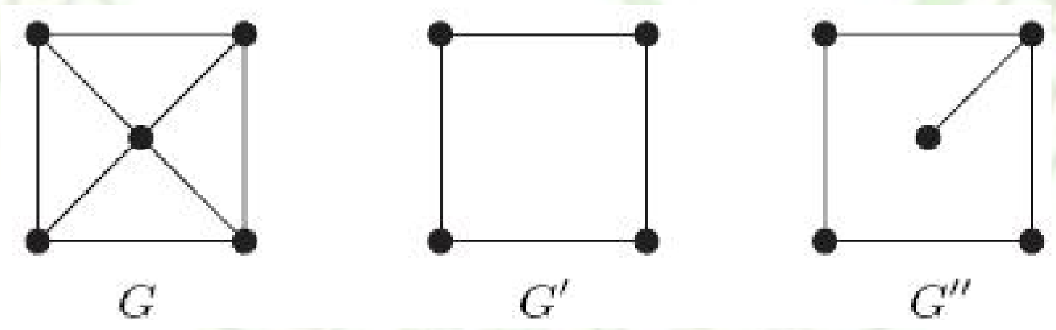
- If $G=(V,E)$ is a graph and $e \in E$, a subgraph G_e is obtained by omitting the edge e from E and keeping all vertices.



G_1 G_2

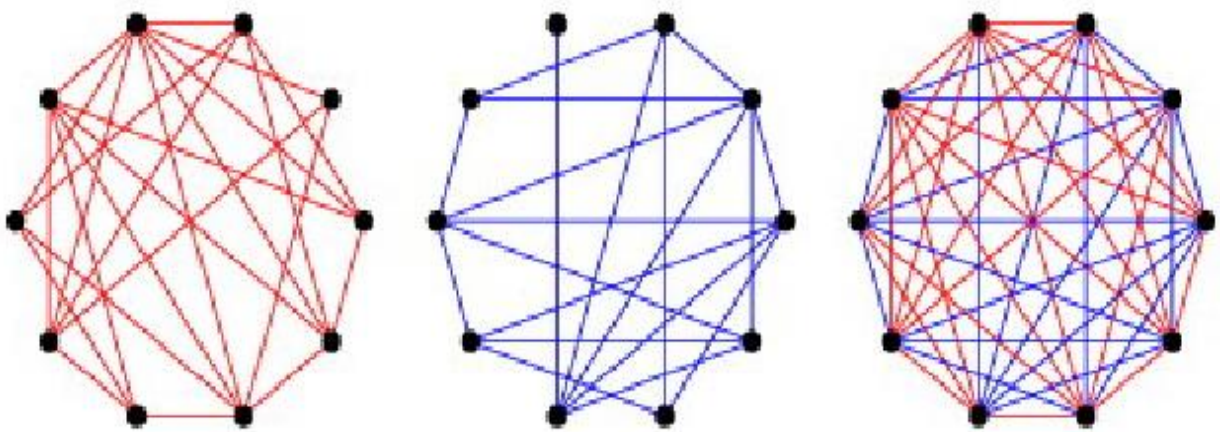
Induced Subgraph

- If $G \subseteq G'$ and G' contains all edges $xy \in E$ with $x, y \in V'$, then G' is an induced subgraph of G. we say that V' induces or spans G' in G, and write $G' := G[V']$.



Graph complement

- The complement of a graph G is the graph G' (sometimes denoted $\overline{G}$) with the same vertex set but whose edge set consists of the edges not present in G.



Graph Unions

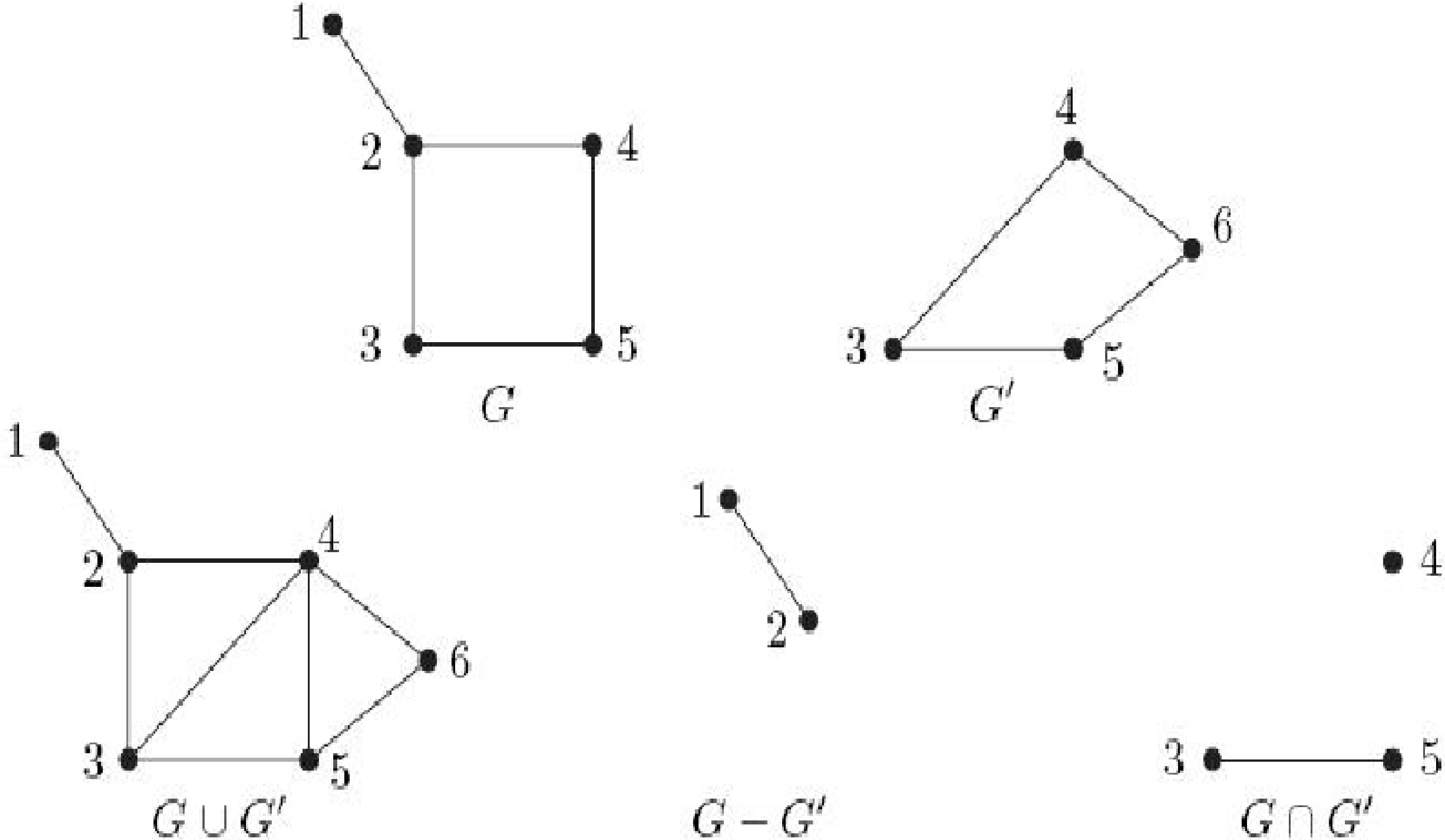
- The *union* $G_1 \cup G_2$ of two simple graphs $G_1=(V_1, E_1)$ and $G_2=(V_2,E_2)$ is the simple graph $(V_1 \cup V_2, E_1 \cup E_2)$.

Graph Intersection

- The intersection $G_1 \cap G_2$ of two simple graphs $G_1=(V_1, E_1)$ and $G_2=(V_2, E_2)$ is the simple graph $(V_1 \cap V_2, E_1 \cap E_2)$.

- The diffrence $G_1 - G_2$ of two simple graphs $G_1=(V_1, E_1)$ and $G_2=(V_2, E_2)$ is the simple graph $(V_1 - V_2, E_1 - E_2)$.

Union,Intersection,diffrence



Homework

- § 10.2
  - 12, 42(参考41和握手定理), 64, 72