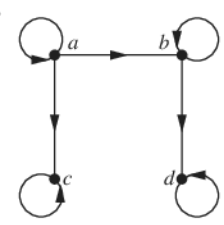
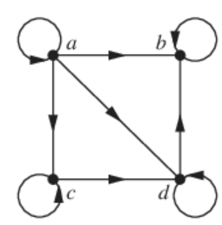
## Exercises

1. Which of these relations on $\{0, 1, 2, 3\}$ are partial orderings? Determine the properties of a partial ordering that the others lack.
   **a)** $\{(0, 0), (1, 1), (2, 2), (3, 3)\}$
   **b)** $\{(0, 0), (1, 1), (2, 0), (2, 2), (2, 3), (3, 2), (3, 3)\}$
   **c)** $\{(0, 0), (1, 1), (1, 2), (2, 2), (3, 3)\}$
   **d)** $\{(0, 0), (1, 1), (1, 2), (1, 3), (2, 2), (2, 3), (3, 3)\}$
   **e)** $\{(0, 0), (0, 1), (0, 2), (1, 0), (1, 1), (1, 2), (2, 0), (2, 2), (3, 3)\}$
2. Which of these relations on $\{0, 1, 2, 3\}$ are partial orderings? Determine the properties of a partial ordering that the others lack.
   **a)** $\{(0, 0), (2, 2), (3, 3)\}$
   **b)** $\{(0, 0), (1, 1), (2, 0), (2, 2), (2, 3), (3, 3)\}$
   **c)** $\{(0, 0), (1, 1), (1, 2), (2, 2), (3, 1), (3, 3)\}$
   **d)** $\{(0, 0), (1, 1), (1, 2), (1, 3), (2, 0), (2, 2), (2, 3), (3, 0), (3, 3)\}$
   **e)** $\{(0, 0), (0, 1), (0, 2), (0, 3), (1, 0), (1, 1), (1, 2), (1, 3), (2, 0), (2, 2), (3, 3)\}$
3. Is $(S, R)$ a poset if $S$ is the set of all people in the world and $(a, b) \in R$, where $a$ and $b$ are people, if
   **a)** $a$ is taller than $b$?
   **b)** $a$ is not taller than $b$?
   **c)** $a = b$ or $a$ is an ancestor of $b$?
   **d)** $a$ and $b$ have a common friend?
4. Is $(S, R)$ a poset if $S$ is the set of all people in the world and $(a, b) \in R$, where $a$ and $b$ are people, if
   **a)** $a$ is no shorter than $b$?
   **b)** $a$ weighs more than $b$?
   **c)** $a = b$ or $a$ is a descendant of $b$?
   **d)** $a$ and $b$ do not have a common friend?
5. Which of these are posets?
   **a)** $(\mathbf{Z}, =)$ **b)** $(\mathbf{Z}, \neq)$ **c)** $(\mathbf{Z}, \geq)$ **d)** $(\mathbf{Z}, \nmid)$
6. Which of these are posets?
   **a)** $(\mathbf{R}, =)$ **b)** $(\mathbf{R}, <)$ **c)** $(\mathbf{R}, \leq)$ **d)** $(\mathbf{R}, \neq)$
7. Determine whether the relations represented by these zero–one matrices are partial orders.
   **a)** $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ **b)** $\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
   **c)** $\begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$
8. Determine whether the relations represented by these zero–one matrices are partial orders.
   **a)** $\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ **b)** $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$
   **c)** $\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$

In Exercises 9–11 determine whether the relation with the directed graph shown is a partial order.

9.



10.



11.



12. Let $(S, R)$ be a poset. Show that $(S, R^{-1})$ is also a poset, where $R^{-1}$ is the inverse of $R$. The poset $(S, R^{-1})$ is called the **dual** of $(S, R)$.
13. Find the duals of these posets.
   **a)** $(\{0, 1, 2\}, \leq)$ **b)** $(\mathbf{Z}, \geq)$
   **c)** $(P(\mathbf{Z}), \supseteq)$ **d)** $(\mathbf{Z}^+, |)$
14. Which of these pairs of elements are comparable in the poset $(\mathbf{Z}^+, |)$?
   **a)** 5, 15 **b)** 6, 9 **c)** 8, 16 **d)** 7, 7
15. Find two incomparable elements in these posets.
   **a)** $(P(\{0, 1, 2\}), \subseteq)$ **b)** $(\{1, 2, 4, 6, 8\}, |)$
16. Let $S = \{1, 2, 3, 4\}$. With respect to the lexicographic order based on the usual "less than" relation,
   **a)** find all pairs in $S \times S$ less than $(2, 3)$.
   **b)** find all pairs in $S \times S$ greater than $(3, 1)$.
   **c)** draw the Hasse diagram of the poset $(S \times S, \preccurlyeq)$.
17. Find the lexicographic ordering of these $n$-tuples:
   **a)** $(1, 1, 2), (1, 2, 1)$ **b)** $(0, 1, 2, 3), (0, 1, 3, 2)$
   **c)** $(1, 0, 1, 0, 1), (0, 1, 1, 1, 0)$
18. Find the lexicographic ordering of these strings of lowercase English letters:
   **a)** *quack, quick, quicksilver, quicksand, quacking*
   **b)** *open, opener, opera, operand, opened*
   **c)** *zoo, zero, zoom, zoology, zoological*
19. Find the lexicographic ordering of the bit strings 0, 01, 11, 001, 010, 011, 0001, and 0101 based on the ordering $0 < 1$.
20. Draw the Hasse diagram for the "greater than or equal to" relation on $\{0, 1, 2, 3, 4, 5\}$.

**21.** Draw the Hasse diagram for the "less than or equal to" relation on $\{0, 2, 5, 10, 11, 15\}$.

**22.** Draw the Hasse diagram for divisibility on the set

**a)** $\{1, 2, 3, 4, 5, 6\}$. **b)** $\{3, 5, 7, 11, 13, 16, 17\}$.

**c)** $\{2, 3, 5, 10, 11, 15, 25\}$. **d)** $\{1, 3, 9, 27, 81, 243\}$.

**23.** Draw the Hasse diagram for divisibility on the set

**a)** $\{1, 2, 3, 4, 5, 6, 7, 8\}$. **b)** $\{1, 2, 3, 5, 7, 11, 13\}$.

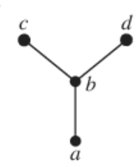
**c)** $\{1, 2, 3, 6, 12, 24, 36, 48\}$.

**d)** $\{1, 2, 4, 8, 16, 32, 64\}$.

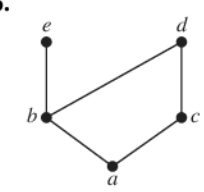
**24.** Draw the Hasse diagram for inclusion on the set $P(S)$, where $S = \{a, b, c, d\}$.

In Exercises 25–27 list all ordered pairs in the partial ordering with the accompanying Hasse diagram.

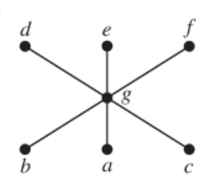
**25.**




**26.**




**27.**



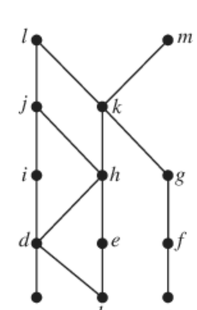

**28.** What is the covering relation of the partial ordering $\{(a, b) \mid a \text{ divides } b\}$ on $\{1, 2, 3, 4, 6, 12\}$?

**29.** What is the covering relation of the partial ordering $\{(A, B) \mid A \subseteq B\}$ on the power set of $S$, where $S = \{a, b, c\}$?

**30.** What is the covering relation of the partial ordering for the poset of security classes defined in Example 25?

**31.** Show that a finite poset can be reconstructed from its covering relation. [*Hint:* Show that the poset is the reflexive transitive closure of its covering relation.]

**32.** Answer these questions for the partial order represented by this Hasse diagram.




**a)** Find the maximal elements.

**b)** Find the minimal elements.

**c)** Is there a greatest element?

**d)** Is there a least element?

**e)** Find all upper bounds of $\{a, b, c\}$.

**f)** Find the least upper bound of $\{a, b, c\}$, if it exists.

**g)** Find all lower bounds of $\{f, g, h\}$.

**h)** Find the greatest lower bound of $\{f, g, h\}$, if it exists.

**33.** Answer these questions for the poset $(\{3, 5, 9, 15, 24, 45\}, |)$.

**a)** Find the maximal elements.

**b)** Find the minimal elements.

**c)** Is there a greatest element?

**d)** Is there a least element?

**e)** Find all upper bounds of $\{3, 5\}$.

**f)** Find the least upper bound of $\{3, 5\}$, if it exists.

**g)** Find all lower bounds of $\{15, 45\}$.

**h)** Find the greatest lower bound of $\{15, 45\}$, if it exists.

**34.** Answer these questions for the poset $(\{2, 4, 6, 9, 12, 18, 27, 36, 48, 60, 72\}, |)$.

**a)** Find the maximal elements.

**b)** Find the minimal elements.

**c)** Is there a greatest element?

**d)** Is there a least element?

**e)** Find all upper bounds of $\{2, 9\}$.

**f)** Find the least upper bound of $\{2, 9\}$, if it exists.

**g)** Find all lower bounds of $\{60, 72\}$.

**h)** Find the greatest lower bound of $\{60, 72\}$, if it exists.

**35.** Answer these questions for the poset $(\{\{1\}, \{2\}, \{4\}, \{1, 2\}, \{1, 4\}, \{2, 4\}, \{3, 4\}, \{1, 3, 4\}, \{2, 3, 4\}\}, \subseteq)$.

**a)** Find the maximal elements.

**b)** Find the minimal elements.

**c)** Is there a greatest element?

**d)** Is there a least element?

**e)** Find all upper bounds of $\{\{2\}, \{4\}\}$.

**f)** Find the least upper bound of $\{\{2\}, \{4\}\}$, if it exists.

**g)** Find all lower bounds of $\{\{1, 3, 4\}, \{2, 3, 4\}\}$.

**h)** Find the greatest lower bound of $\{\{1, 3, 4\}, \{2, 3, 4\}\}$, if it exists.

**36.** Give a poset that has

**a)** a minimal element but no maximal element.

**b)** a maximal element but no minimal element.

**c)** neither a maximal nor a minimal element.

**37.** Show that lexicographic order is a partial ordering on the Cartesian product of two posets.

**38.** Show that lexicographic order is a partial ordering on the set of strings from a poset.

**39.** Suppose that $(S, \preccurlyeq_1)$ and $(T, \preccurlyeq_2)$ are posets. Show that $(S \times T, \preccurlyeq)$ is a poset where $(s, t) \preccurlyeq (u, v)$ if and only if $s \preccurlyeq_1 u$ and $t \preccurlyeq_2 v$.

**40.** **a)** Show that there is exactly one greatest element of a poset, if such an element exists.

**b)** Show that there is exactly one least element of a poset, if such an element exists.

**41.** **a)** Show that there is exactly one maximal element in a poset with a greatest element.

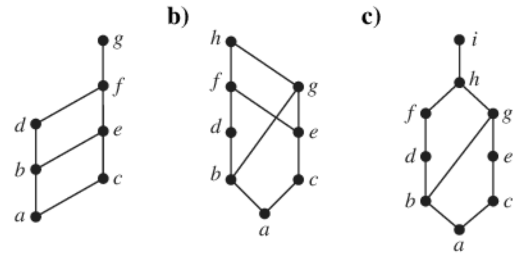
**b)** Show that there is exactly one minimal element in a poset with a least element.

**42.** **a)** Show that the least upper bound of a set in a poset is unique if it exists.

**b)** Show that the greatest lower bound of a set in a poset is unique if it exists.

**43.** Determine whether the posets with these Hasse diagrams are lattices.

**a)** **b)** **c)**




**44.** Determine whether these posets are lattices.

**a)** $(\{1, 3, 6, 9, 12\}, |)$ **b)** $(\{1, 5, 25, 125\}, |)$

**c)** $(\mathbf{Z}, \geq)$

**d)** $(P(S), \supseteq)$, where $P(S)$ is the power set of a set $S$

**45.** Show that every nonempty finite subset of a lattice has a least upper bound and a greatest lower bound.

**46.** Show that if the poset $(S, R)$ is a lattice then the dual poset $(S, R^{-1})$ is also a lattice.

**47.** In a company, the lattice model of information flow is used to control sensitive information with security classes represented by ordered pairs $(A, C)$. Here $A$ is an authority level, which may be nonproprietary (0), proprietary (1), restricted (2), or registered (3). A category $C$ is a subset of the set of all projects {*Cheetah, Impala, Puma*}. (Names of animals are often used as code names for projects in companies.)

**a)** Is information permitted to flow from (*Proprietary*, {*Cheetah, Puma*}) into (*Restricted*, {*Puma*})?

**b)** Is information permitted to flow from (*Restricted*, {*Cheetah*}) into (*Registered*, {*Cheetah, Impala*})?

**c)** Into which classes is information from (*Proprietary*, {*Cheetah, Puma*}) permitted to flow?

**d)** From which classes is information permitted to flow into the security class (*Restricted*, {*Impala, Puma*})?

**48.** Show that the set $S$ of security classes $(A, C)$ is a lattice, where $A$ is a positive integer representing an authority class and $C$ is a subset of a finite set of compartments, with $(A_1, C_1) \preccurlyeq (A_2, C_2)$ if and only if $A_1 \leq A_2$ and $C_1 \subseteq C_2$. [*Hint:* First show that $(S, \preccurlyeq)$ is a poset and then show that the least upper bound and greatest lower bound of $(A_1, C_1)$ and $(A_2, C_2)$ are $(\max(A_1, A_2), C_1 \cup C_2)$ and $(\min(A_1, A_2), C_1 \cap C_2)$, respectively.]

***49.** Show that the set of all partitions of a set $S$ with the relation $P_1 \preccurlyeq P_2$ if the partition $P_1$ is a refinement of the partition $P_2$ is a lattice. (See the preamble to Exercise 49 of Section 9.5.)

**50.** Show that every totally ordered set is a lattice.

**51.** Show that every finite lattice has a least element and a greatest element.

**52.** Give an example of an infinite lattice with

**a)** neither a least nor a greatest element.

**b)** a least but not a greatest element.

**c)** a greatest but not a least element.

**d)** both a least and a greatest element.

**53.** Verify that $(\mathbf{Z}^+ \times \mathbf{Z}^+, \preccurlyeq)$ is a well-ordered set, where $\preccurlyeq$ is lexicographic order, as claimed in Example 8.

**54.** Determine whether each of these posets is well-ordered.

**a)** $(S, \leq)$, where $S = \{10, 11, 12, \ldots\}$

**b)** $(\mathbf{Q} \cap [0, 1], \leq)$ (the set of rational numbers between 0 and 1 inclusive)

**c)** $(S, \leq)$, where $S$ is the set of positive rational numbers with denominators not exceeding 3

**d)** $(\mathbf{Z}^-, \geq)$, where $\mathbf{Z}^-$ is the set of negative integers

A poset $(R, \preccurlyeq)$ is **well-founded** if there is no infinite decreasing sequence of elements in the poset, that is, elements $x_1, x_2, \ldots, x_n$ such that $\cdots \prec x_n \prec \cdots \prec x_2 \prec x_1$. A poset $(R, \preccurlyeq)$ is **dense** if for all $x \in S$ and $y \in S$ with $x \prec y$, there is an element $z \in R$ such that $x \prec z \prec y$.

**55.** Show that the poset $(\mathbf{Z}, \preccurlyeq)$, where $x \prec y$ if and only if $|x| < |y|$ is well-founded but is not a totally ordered set.

**56.** Show that a dense poset with at least two elements that are comparable is not well-founded.

**57.** Show that the poset of rational numbers with the usual "less than or equal to" relation, $(\mathbf{Q}, \leq)$, is a dense poset.

***58.** Show that the set of strings of lowercase English letters with lexicographic order is neither well-founded nor dense.

**59.** Show that a poset is well-ordered if and only if it is totally ordered and well-founded.

**60.** Show that a finite nonempty poset has a maximal element.

**61.** Find a compatible total order for the poset with the Hasse diagram shown in Exercise 32.

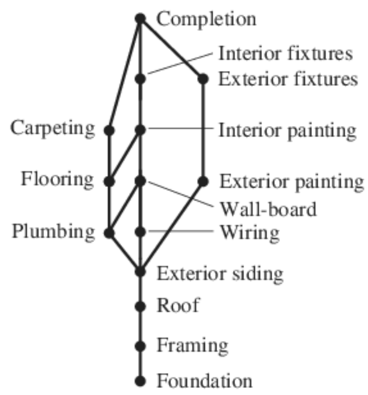
**62.** Find a compatible total order for the divisibility relation on the set $\{1, 2, 3, 6, 8, 12, 24, 36\}$.

**63.** Find all compatible total orderings for the poset $(\{1, 2, 4, 5, 12, 20\}, |\}$ from Example 26.

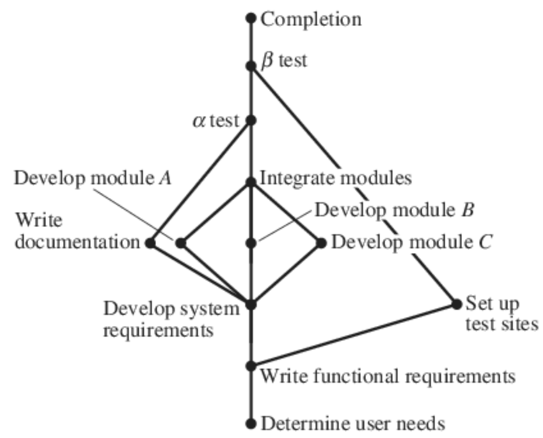
**64.** Find all compatible total orderings for the poset with the Hasse diagram in Exercise 27.

**65.** Find all possible orders for completing the tasks in the development project in Example 27.

**66.** Schedule the tasks needed to build a house, by specifying their order, if the Hasse diagram representing these tasks is as shown in the figure.




**67.** Find an ordering of the tasks of a software project if the Hasse diagram for the tasks of the project is as shown.




## Key Terms and Results

### TERMS

**binary relation from $A$ to $B$:** a subset of $A \times B$

**relation on $A$:** a binary relation from $A$ to itself (i.e., a subset of $A \times A$)

**$S \circ R$:** composite of $R$ and $S$

**$R^{-1}$:** inverse relation of $R$

**$R^n$:** $n$th power of $R$

**reflexive:** a relation $R$ on $A$ is reflexive if $(a, a) \in R$ for all $a \in A$

**symmetric:** a relation $R$ on $A$ is symmetric if $(b, a) \in R$ whenever $(a, b) \in R$

**antisymmetric:** a relation $R$ on $A$ is antisymmetric if $a = b$ whenever $(a, b) \in R$ and $(b, a) \in R$

**transitive:** a relation $R$ on $A$ is transitive if $(a, b) \in R$ and $(b, c) \in R$ implies that $(a, c) \in R$

**$n$-ary relation on $A_1, A_2, \ldots, A_n$:** a subset of $A_1 \times A_2 \times \cdots \times A_n$

**relational data model:** a model for representing databases using $n$-ary relations

**primary key:** a domain of an $n$-ary relation such that an $n$-tuple is uniquely determined by its value for this domain

**composite key:** the Cartesian product of domains of an $n$-ary relation such that an $n$-tuple is uniquely determined by its values in these domains

**selection operator:** a function that selects the $n$-tuples in an $n$-ary relation that satisfy a specified condition

**projection:** a function that produces relations of smaller degree from an $n$-ary relation by deleting fields

**join:** a function that combines $n$-ary relations that agree on certain fields

**directed graph or digraph:** a set of elements called vertices and ordered pairs of these elements, called edges

**loop:** an edge of the form $(a, a)$

**closure of a relation $R$ with respect to a property P:** the relation $S$ (if it exists) that contains $R$, has property **P**, and is contained within any relation that contains $R$ and has property **P**

**path in a digraph:** a sequence of edges $(a, x_1), (x_1, x_2), \ldots, (x_{n-2}, x_{n-1}), (x_{n-1}, b)$ such that the terminal vertex of each edge is the initial vertex of the succeeding edge in the sequence

**circuit (or cycle) in a digraph:** a path that begins and ends at the same vertex

**$R^*$ (connectivity relation):** the relation consisting of those ordered pairs $(a, b)$ such that there is a path from $a$ to $b$

**equivalence relation:** a reflexive, symmetric, and transitive relation

**equivalent:** if $R$ is an equivalence relation, $a$ is equivalent to $b$ if $aRb$

**$[a]_R$ (equivalence class of $a$ with respect to $R$):** the set of all elements of $A$ that are equivalent to $a$

**$[a]_m$ (congruence class modulo $m$):** the set of integers congruent to $a$ modulo $m$

**partition of a set $S$:** a collection of pairwise disjoint nonempty subsets that have $S$ as their union

**partial ordering:** a relation that is reflexive, antisymmetric, and transitive

**poset $(S, R)$:** a set $S$ and a partial ordering $R$ on this set

**comparable:** the elements $a$ and $b$ in the poset $(A, \preccurlyeq)$ are comparable if $a \preccurlyeq b$ or $b \preccurlyeq a$

**incomparable:** elements in a poset that are not comparable

**total (or linear) ordering:** a partial ordering for which every pair of elements are comparable

**totally (or linearly) ordered set:** a poset with a total (or linear) ordering

**well-ordered set:** a poset $(S, \preccurlyeq)$, where $\preccurlyeq$ is a total order and every nonempty subset of $S$ has a least element

**lexicographic order:** a partial ordering of Cartesian products or strings

**Hasse diagram:** a graphical representation of a poset where loops and all edges resulting from the transitive property are not shown, and the direction of the edges is indicated by the position of the vertices

**maximal element:** an element of a poset that is not less than any other element of the poset

**minimal element:** an element of a poset that is not greater than any other element of the poset

**greatest element:** an element of a poset greater than all other elements in this set

**least element:** an element of a poset less than all other elements in this set

**upper bound of a set:** an element in a poset greater than all other elements in the set

**lower bound of a set:** an element in a poset less than all other elements in the set

**least upper bound of a set:** an upper bound of the set that is less than all other upper bounds

**greatest lower bound of a set:** a lower bound of the set that is greater than all other lower bounds

**lattice:** a partially ordered set in which every two elements have a greatest lower bound and a least upper bound

**compatible total ordering for a partial ordering:** a total ordering that contains the given partial ordering

**topological sort:** the construction of a total ordering compatible with a given partial ordering

### RESULTS

The reflexive closure of a relation $R$ on the set $A$ equals $R \cup \Delta$, where $\Delta = \{(a, a) \mid a \in A\}$.

The symmetric closure of a relation $R$ on the set $A$ equals $R \cup R^{-1}$, where $R^{-1} = \{(b, a) \mid (a, b) \in R\}$.

The transitive closure of a relation equals the connectivity relation formed from this relation.

Warshall's algorithm for finding the transitive closure of a relation

Let $R$ be an equivalence relation. Then the following three statements are equivalent: (1) $a\,R\,b$; (2) $[a]_R \cap [b]_R \neq \emptyset$; (3) $[a]_R = [b]_R$.

The equivalence classes of an equivalence relation on a set $A$ form a partition of $A$. Conversely, an equivalence relation can be constructed from any partition so that the equivalence classes are the subsets in the partition.

The principle of well-ordered induction

The topological sorting algorithm

## Review Questions

**1. a)** What is a relation on a set?

**b)** How many relations are there on a set with $n$ elements?

**2. a)** What is a reflexive relation?

**b)** What is a symmetric relation?

**c)** What is an antisymmetric relation?

**d)** What is a transitive relation?

**3.** Give an example of a relation on the set $\{1, 2, 3, 4\}$ that is

**a)** reflexive, symmetric, and not transitive.

**b)** not reflexive, symmetric, and transitive.

**c)** reflexive, antisymmetric, and not transitive.

**d)** reflexive, symmetric, and transitive.

**e)** reflexive, antisymmetric, and transitive.

**4. a)** How many reflexive relations are there on a set with $n$ elements?

**b)** How many symmetric relations are there on a set with $n$ elements?

**c)** How many antisymmetric relations are there on a set with $n$ elements?

**5. a)** Explain how an $n$-ary relation can be used to represent information about students at a university.

**b)** How can the 5-ary relation containing names of students, their addresses, telephone numbers, majors, and grade point averages be used to form a 3-ary relation containing the names of students, their majors, and their grade point averages?

**c)** How can the 4-ary relation containing names of students, their addresses, telephone numbers, and majors and the 4-ary relation containing names of students, their student numbers, majors, and numbers of credit hours be combined into a single $n$-ary relation?

**6. a)** Explain how to use a zero–one matrix to represent a relation on a finite set.

**b)** Explain how to use the zero–one matrix representing a relation to determine whether the relation is reflexive, symmetric, and/or antisymmetric.

**7. a)** Explain how to use a directed graph to represent a relation on a finite set.

**b)** Explain how to use the directed graph representing a relation to determine whether a relation is reflexive, symmetric, and/or antisymmetric.

**8. a)** Define the reflexive closure and the symmetric closure of a relation.

**b)** How can you construct the reflexive closure of a relation?

**c)** How can you construct the symmetric closure of a relation?

**d)** Find the reflexive closure and the symmetric closure of the relation $\{(1, 2), (2, 3), (2, 4), (3, 1)\}$ on the set $\{1, 2, 3, 4\}$.

**9. a)** Define the transitive closure of a relation.

**b)** Can the transitive closure of a relation be obtained by including all pairs $(a, c)$ such that $(a, b)$ and $(b, c)$ belong to the relation?

c) Describe two algorithms for finding the transitive closure of a relation.

d) Find the transitive closure of the relation $\{(1,1), (1,3), (2,1), (2,3), (2,4), (3,2), (3,4), (4,1)\}$.

**10. a)** Define an equivalence relation.

**b)** Which relations on the set $\{a, b, c, d\}$ are equivalence relations and contain $(a, b)$ and $(b, d)$?

**11. a)** Show that congruence modulo $m$ is an equivalence relation whenever $m$ is a positive integer.

**b)** Show that the relation $\{(a, b) \mid a \equiv \pm b \pmod 7\}$ is an equivalence relation on the set of integers.

**12. a)** What are the equivalence classes of an equivalence relation?

**b)** What are the equivalence classes of the "congruent modulo 5" relation?

**c)** What are the equivalence classes of the equivalence relation in Question 11(b)?

**13.** Explain the relationship between equivalence relations on a set and partitions of this set.

**14. a)** Define a partial ordering.

**b)** Show that the divisibility relation on the set of positive integers is a partial order.

**15.** Explain how partial orderings on the sets $A_1$ and $A_2$ can be used to define a partial ordering on the set $A_1 \times A_2$.

**16. a)** Explain how to construct the Hasse diagram of a partial order on a finite set.

**b)** Draw the Hasse diagram of the divisibility relation on the set $\{2, 3, 5, 9, 12, 15, 18\}$.

**17. a)** Define a maximal element of a poset and the greatest element of a poset.

**b)** Give an example of a poset that has three maximal elements.

**c)** Give an example of a poset with a greatest element.

**18. a)** Define a lattice.

**b)** Give an example of a poset with five elements that is a lattice and an example of a poset with five elements that is not a lattice.

**19. a)** Show that every finite subset of a lattice has a greatest lower bound and a least upper bound.

**b)** Show that every lattice with a finite number of elements has a least element and a greatest element.

**20. a)** Define a well-ordered set.

**b)** Describe an algorithm for producing a totally ordered set compatible with a given partially ordered set.

**c)** Explain how the algorithm from (b) can be used to order the tasks in a project if tasks are done one at a time and each task can be done only after one or more of the other tasks have been completed.

## Supplementary Exercises

**1.** Let $S$ be the set of all strings of English letters. Determine whether these relations are reflexive, irreflexive, symmetric, antisymmetric, and/or transitive.

**a)** $R_1 = \{(a, b) \mid a \text{ and } b \text{ have no letters in common}\}$

**b)** $R_2 = \{(a, b) \mid a \text{ and } b \text{ are not the same length}\}$

**c)** $R_3 = \{(a, b) \mid a \text{ is longer than } b\}$

**2.** Construct a relation on the set $\{a, b, c, d\}$ that is

**a)** reflexive, symmetric, but not transitive.

**b)** irreflexive, symmetric, and transitive.

**c)** irreflexive, antisymmetric, and not transitive.

**d)** reflexive, neither symmetric nor antisymmetric, and transitive.

**e)** neither reflexive, irreflexive, symmetric, antisymmetric, nor transitive.

**3.** Show that the relation $R$ on $\mathbf{Z} \times \mathbf{Z}$ defined by $(a, b)\, R\, (c, d)$ if and only if $a + d = b + c$ is an equivalence relation.

**4.** Show that a subset of an antisymmetric relation is also antisymmetric.

**5.** Let $R$ be a reflexive relation on a set $A$. Show that $R \subseteq R^2$.

**6.** Suppose that $R_1$ and $R_2$ are reflexive relations on a set $A$. Show that $R_1 \oplus R_2$ is irreflexive.

**7.** Suppose that $R_1$ and $R_2$ are reflexive relations on a set $A$. Is $R_1 \cap R_2$ also reflexive? Is $R_1 \cup R_2$ also reflexive?

**8.** Suppose that $R$ is a symmetric relation on a set $A$. Is $\overline{R}$ also symmetric?

**9.** Let $R_1$ and $R_2$ be symmetric relations. Is $R_1 \cap R_2$ also symmetric? Is $R_1 \cup R_2$ also symmetric?

**10.** A relation $R$ is called **circular** if $aRb$ and $bRc$ imply that $cRa$. Show that $R$ is reflexive and circular if and only if it is an equivalence relation.

**11.** Show that a primary key in an $n$-ary relation is a primary key in any projection of this relation that contains this key as one of its fields.

**12.** Is the primary key in an $n$-ary relation also a primary key in a larger relation obtained by taking the join of this relation with a second relation?

**13.** Show that the reflexive closure of the symmetric closure of a relation is the same as the symmetric closure of its reflexive closure.

**14.** Let $R$ be the relation on the set of all mathematicians that contains the ordered pair $(a, b)$ if and only if $a$ and $b$ have written a published mathematical paper together.

**a)** Describe the relation $R^2$.

**b)** Describe the relation $R^*$.

**c)** The **Erdős number** of a mathematician is 1 if this mathematician wrote a paper with the prolific Hungarian mathematician Paul Erdős, it is 2 if this mathematician did not write a joint paper with Erdős but wrote a joint paper with someone who wrote a joint paper with Erdős, and so on (except that the Erdős number of Erdős himself is 0). Give a definition of the Erdős number in terms of paths in $R$.

**15.** **a)** Give an example to show that the transitive closure of the symmetric closure of a relation is not necessarily the same as the symmetric closure of the transitive closure of this relation.

**b)** Show, however, that the transitive closure of the symmetric closure of a relation must contain the symmetric closure of the transitive closure of this relation.

**16.** **a)** Let $S$ be the set of subroutines of a computer program. Define the relation $R$ by $\mathbf{P} R \mathbf{Q}$ if subroutine **P** calls subroutine **Q** during its execution. Describe the transitive closure of $R$.

**b)** For which subroutines **P** does $(\mathbf{P}, \mathbf{P})$ belong to the transitive closure of $R$?

**c)** Describe the reflexive closure of the transitive closure of $R$.

**17.** Suppose that $R$ and $S$ are relations on a set $A$ with $R \subseteq S$ such that the closures of $R$ and $S$ with respect to a property **P** both exist. Show that the closure of $R$ with respect to **P** is a subset of the closure of $S$ with respect to **P**.

**18.** Show that the symmetric closure of the union of two relations is the union of their symmetric closures.

***19.** Devise an algorithm, based on the concept of interior vertices, that finds the length of the longest path between two vertices in a directed graph, or determines that there are arbitrarily long paths between these vertices.

**20.** Which of these are equivalence relations on the set of all people?

**a)** $\{(x, y) \mid x$ and $y$ have the same sign of the zodiac$\}$

**b)** $\{(x, y) \mid x$ and $y$ were born in the same year$\}$

**c)** $\{(x, y) \mid x$ and $y$ have been in the same city$\}$

***21.** How many different equivalence relations with exactly three different equivalence classes are there on a set with five elements?

**22.** Show that $\{(x, y) \mid x - y \in \mathbf{Q}\}$ is an equivalence relation on the set of real numbers, where **Q** denotes the set of rational numbers. What are $[1]$, $[\frac{1}{2}]$, and $[\pi]$?

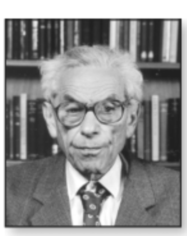
**23.** Suppose that $P_1 = \{A_1, A_2, \ldots, A_m\}$ and $P_2 = \{B_1, B_2, \ldots, B_n\}$ are both partitions of the set $S$. Show that the collection of nonempty subsets of the form $A_i \cap B_j$ is a partition of $S$ that is a refinement of both $P_1$ and $P_2$ (see the preamble to Exercise 49 of Section 9.5).

***24.** Show that the transitive closure of the symmetric closure of the reflexive closure of a relation $R$ is the smallest equivalence relation that contains $R$.

**25.** Let $\mathbf{R}(S)$ be the set of all relations on a set $S$. Define the relation $\preccurlyeq$ on $\mathbf{R}(S)$ by $R_1 \preccurlyeq R_2$ if $R_1 \subseteq R_2$, where $R_1$ and $R_2$ are relations on $S$. Show that $(\mathbf{R}(S), \preccurlyeq)$ is a poset.

**26.** Let $\mathbf{P}(S)$ be the set of all partitions of the set $S$. Define the relation $\preccurlyeq$ on $\mathbf{P}(S)$ by $P_1 \preccurlyeq P_2$ if $P_1$ is a refinement of $P_2$ (see Exercise 49 of Section 9.5). Show that $(\mathbf{P}(S), \preccurlyeq)$ is a poset.

Links



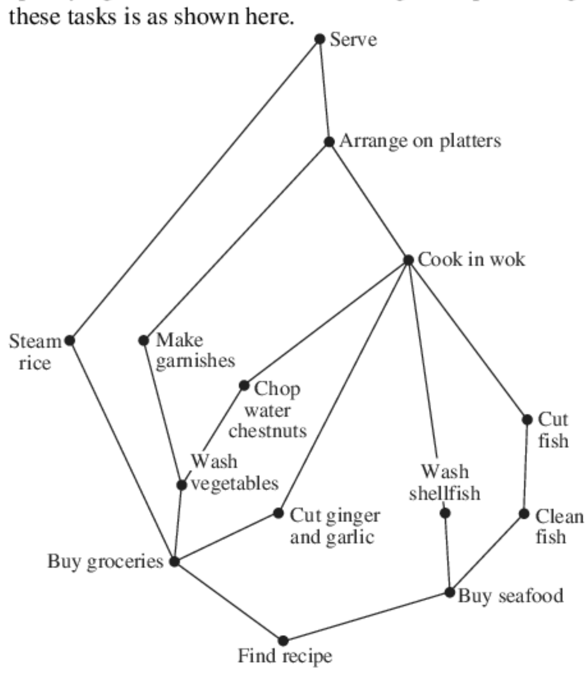
PAUL ERDŐS (1913–1996) Paul Erdős, born in Budapest, Hungary, was the son of two high school mathematics teachers. He was a child prodigy; at age 3 he could multiply three-digit numbers in his head, and at 4 he discovered negative numbers on his own. Because his mother did not want to expose him to contagious diseases, he was mostly home-schooled. At 17 Erdős entered Eőtvős University, graduating four years later with a Ph.D. in mathematics. After graduating he spent four years at Manchester, England, on a postdoctoral fellowship. In 1938 he went to the United States because of the difficult political situation in Hungary, especially for Jews. He spent much of his time in the United States, except for 1954 to 1962, when he was banned as part of the paranoia of the McCarthy era. He also spent considerable time in Israel.

Erdős made many significant contributions to combinatorics and to number theory. One of the discoveries of which he was most proud is his elementary proof (in the sense that it does not use any complex analysis) of the prime number theorem, which provides an estimate for the number of primes not exceeding a fixed positive integer. He also participated in the modern development of the Ramsey theory.

Erdős traveled extensively throughout the world to work with other mathematicians, visiting conferences, universities, and research laboratories. He had no permanent home. He devoted himself almost entirely to mathematics, traveling from one mathematician to the next, proclaiming "My brain is open." Erdős was the author or coauthor of more than 1500 papers and had more than 500 coauthors. Copies of his articles are kept by Ron Graham, a famous discrete mathematician with whom he collaborated extensively and who took care of many of his worldly needs.

Erdős offered rewards, ranging from \$10 to \$10,000, for the solution of problems that he found particularly interesting, with the size of the reward depending on the difficulty of the problem. He paid out close to \$4000. Erdős had his own special language, using such terms as "epsilon" (child), "boss" (woman), "slave" (man), "captured" (married), "liberated" (divorced), "Supreme Fascist" (God), "Sam" (United States), and "Joe" (Soviet Union). Although he was curious about many things, he concentrated almost all his energy on mathematical research. He had no hobbies and no full-time job. He never married and apparently remained celibate. Erdős was extremely generous, donating much of the money he collected from prizes, awards, and stipends for scholarships and to worthwhile causes. He traveled extremely lightly and did not like having many material possessions.

**27.** Schedule the tasks needed to cook a Chinese meal by specifying their order, if the Hasse diagram representing these tasks is as shown here.




A subset of a poset such that every two elements of this subset are comparable is called a **chain**. A subset of a poset is called an **antichain** if every two elements of this subset are incomparable.

**28.** Find all chains in the posets with the Hasse diagrams shown in Exercises 25–27 in Section 9.6.

**29.** Find all antichains in the posets with the Hasse diagrams shown in Exercises 25–27 in Section 9.6.

**30.** Find an antichain with the greatest number of elements in the poset with the Hasse diagram of Exercise 32 in Section 9.6.

**31.** Show that every maximal chain in a finite poset $(S, \preccurlyeq)$ contains a minimal element of $S$. (A maximal chain is a chain that is not a subset of a larger chain.)

****32.** Show that every finite poset can be partitioned into $k$ chains, where $k$ is the largest number of elements in an antichain in this poset.

***33.** Show that in any group of $mn + 1$ people there is either a list of $m + 1$ people where a person in the list (except for the first person listed) is a descendant of the previous person on the list, or there are $n + 1$ people such that none of these people is a descendant of any of the other $n$ people. [*Hint:* Use Exercise 32.]

Suppose that $(S, \preccurlyeq)$ is a well-founded partially ordered set. The *principle of well-founded induction* states that $P(x)$ is true for all $x \in S$ if $\forall x(\forall y(y \prec x \rightarrow P(y)) \rightarrow P(x))$.

**34.** Show that no separate basis case is needed for the principle of well-founded induction. That is, $P(u)$ is true for all minimal elements $u$ in $S$ if $\forall x(\forall y(y \prec x \rightarrow P(y)) \rightarrow P(x))$.

***35.** Show that the principle of well-founded induction is valid.

A relation $R$ on a set $A$ is a **quasi-ordering** on $A$ if $R$ is reflexive and transitive.

**36.** Let $R$ be the relation on the set of all functions from $\mathbf{Z}^+$ to $\mathbf{Z}^+$ such that $(f, g)$ belongs to $R$ if and only if $f$ is $O(g)$. Show that $R$ is a quasi-ordering.

**37.** Let $R$ be a quasi-ordering on a set $A$. Show that $R \cap R^{-1}$ is an equivalence relation.

***38.** Let $R$ be a quasi-ordering and let $S$ be the relation on the set of equivalence classes of $R \cap R^{-1}$ such that $(C, D)$ belongs to $S$, where $C$ and $D$ are equivalence classes of $R$, if and only if there are elements $c$ of $C$ and $d$ of $D$ such that $(c, d)$ belongs to $R$. Show that $S$ is a partial ordering.

Let $L$ be a lattice. Define the **meet** ($\wedge$) and **join** ($\vee$) operations by $x \wedge y = \text{glb}(x, y)$ and $x \vee y = \text{lub}(x, y)$.

**39.** Show that the following properties hold for all elements $x$, $y$, and $z$ of a lattice $L$.

**a)** $x \wedge y = y \wedge x$ and $x \vee y = y \vee x$ **(commutative laws)**

**b)** $(x \wedge y) \wedge z = x \wedge (y \wedge z)$ and $(x \vee y) \vee z = x \vee (y \vee z)$ **(associative laws)**

**c)** $x \wedge (x \vee y) = x$ and $x \vee (x \wedge y) = x$ **(absorption laws)**

**d)** $x \wedge x = x$ and $x \vee x = x$ **(idempotent laws)**

**40.** Show that if $x$ and $y$ are elements of a lattice $L$, then $x \vee y = y$ if and only if $x \wedge y = x$.

A lattice $L$ is **bounded** if it has both an **upper bound**, denoted by 1, such that $x \preccurlyeq 1$ for all $x \in L$ and a **lower bound**, denoted by 0, such that $0 \preccurlyeq x$ for all $x \in L$.

**41.** Show that if $L$ is a bounded lattice with upper bound 1 and lower bound 0 then these properties hold for all elements $x \in L$.

**a)** $x \vee 1 = 1$ **b)** $x \wedge 1 = x$

**c)** $x \vee 0 = x$ **d)** $x \wedge 0 = 0$

**42.** Show that every finite lattice is bounded.

A lattice is called **distributive** if $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$ and $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ for all $x$, $y$, and $z$ in $L$.

***43.** Give an example of a lattice that is not distributive.

**44.** Show that the lattice $(P(S), \subseteq)$ where $P(S)$ is the power set of a finite set $S$ is distributive.

**45.** Is the lattice $(\mathbf{Z}^+, |)$ distributive?

The **complement** of an element $a$ of a bounded lattice $L$ with upper bound 1 and lower bound 0 is an element $b$ such that $a \vee b = 1$ and $a \wedge b = 0$. Such a lattice is **complemented** if every element of the lattice has a complement.

**46.** Give an example of a finite lattice where at least one element has more than one complement and at least one element has no complement.

**47.** Show that the lattice $(P(S), \subseteq)$ where $P(S)$ is the power set of a finite set $S$ is complemented.

*48. Show that if $L$ is a finite distributive lattice, then an element of $L$ has at most one complement.

The game of Chomp, introduced in Example 12 in Section 1.8, can be generalized for play on any finite partially ordered set $(S, \preceq)$ with a least element $a$. In this game, a move consists of selecting an element $x$ in $S$ and removing $x$ and all elements larger than it from $S$. The loser is the player who is forced to select the least element $a$.

49. Show that the game of Chomp with cookies arranged in an $m \times n$ rectangular grid, described in Example 12 in Section 1.8, is the same as the game of Chomp on the poset $(S, |)$, where $S$ is the set of all positive integers that divide $p^{m-1}q^{n-1}$, where $p$ and $q$ are distinct primes.

50. Show that if $(S, \preceq)$ has a greatest element $b$, then a winning strategy for Chomp on this poset exists. [*Hint:* Generalize the argument in Example 12 in Section 1.8.]

## Computer Projects

### Write programs with these input and output.

1. Given the matrix representing a relation on a finite set, determine whether the relation is reflexive and/or irreflexive.
2. Given the matrix representing a relation on a finite set, determine whether the relation is symmetric and/or antisymmetric.
3. Given the matrix representing a relation on a finite set, determine whether the relation is transitive.
4. Given a positive integer $n$, display all the relations on a set with $n$ elements.

*5. Given a positive integer $n$, determine the number of transitive relations on a set with $n$ elements.

*6. Given a positive integer $n$, determine the number of equivalence relations on a set with $n$ elements.

*7. Given a positive integer $n$, display all the equivalence relations on the set of the $n$ smallest positive integers.

8. Given an $n$-ary relation, find the projection of this relation when specified fields are deleted.
9. Given an $m$-ary relation and an $n$-ary relation, and a set of common fields, find the join of these relations with respect to these common fields.
10. Given the matrix representing a relation on a finite set, find the matrix representing the reflexive closure of this relation.
11. Given the matrix representing a relation on a finite set, find the matrix representing the symmetric closure of this relation.
12. Given the matrix representing a relation on a finite set, find the matrix representing the transitive closure of this relation by computing the join of the Boolean powers of the matrix representing the relation.
13. Given the matrix representing a relation on a finite set, find the matrix representing the transitive closure of this relation using Warshall's algorithm.
14. Given the matrix representing a relation on a finite set, find the matrix representing the smallest equivalence relation containing this relation.
15. Given a partial ordering on a finite set, find a total ordering compatible with it using topological sorting.

## Computations and Explorations

### Use a computational program or programs you have written to do these exercises.

1. Display all the different relations on a set with four elements.
2. Display all the different reflexive and symmetric relations on a set with six elements.
3. Display all the reflexive and transitive relations on a set with five elements.

*4. Determine how many transitive relations there are on a set with $n$ elements for all positive integers $n$ with $n \leq 7$.

5. Find the transitive closure of a relation of your choice on a set with at least 20 elements. Either use a relation that corresponds to direct links in a particular transportation or communications network or use a randomly generated relation.
6. Compute the number of different equivalence relations on a set with $n$ elements for all positive integers $n$ not exceeding 20.
7. Display all the equivalence relations on a set with seven elements.

*8. Display all the partial orders on a set with five elements.

*9. Display all the lattices on a set with five elements.

## Writing Projects

**Respond to these with essays using outside sources.**

1. Discuss the concept of a fuzzy relation. How are fuzzy relations used?
2. Describe the basic principles of relational databases, going beyond what was covered in Section 9.2. How widely used are relational databases as compared with other types of databases?
3. Look up the original papers by Warshall and by Roy (in French) in which they develop algorithms for finding transitive closures. Discuss their approaches. Why do you suppose that what we call Warshall's algorithm was discovered independently by more than one person?
4. Describe how equivalence classes can be used to define the rational numbers as classes of pairs of integers and how the basic arithmetic operations on rational numbers can be defined following this approach. (See Exercise 40 in Section 9.5.)
5. Explain how Helmut Hasse used what we now call Hasse diagrams.
6. Describe some of the mechanisms used to enforce information flow policies in computer operating systems.
7. Discuss the use of the Program Evaluation and Review Technique (PERT) to schedule the tasks of a large complicated project. How widely is PERT used?
8. Discuss the use of the Critical Path Method (CPM) to find the shortest time for the completion of a project. How widely is CPM used?
9. Discuss the concept of *duality* in a lattice. Explain how duality can be used to establish new results.
10. Explain what is meant by a *modular lattice*. Describe some of the properties of modular lattices and describe how modular lattices arise in the study of projective geometry.

CHAPTER

# 10 Graphs



Graphs are discrete structures consisting of vertices and edges that connect these vertices. There are different kinds of graphs, depending on whether edges have directions, whether multiple edges can connect the same pair of vertices, and whether loops are allowed. Problems in almost every conceivable discipline can be solved using graph models. We will give examples to illustrate how graphs are used as models in a variety of areas. For instance, we will show how graphs are used to represent the competition of different species in an ecological niche, how graphs are used to represent who influences whom in an organization, and how graphs are used to represent the outcomes of round-robin tournaments. We will describe how graphs can be used to model acquaintanceships between people, collaboration between researchers, telephone calls between telephone numbers, and links between websites. We will show how graphs can be used to model roadmaps and the assignment of jobs to employees of an organization.

Using graph models, we can determine whether it is possible to walk down all the streets in a city without going down a street twice, and we can find the number of colors needed to color the regions of a map. Graphs can be used to determine whether a circuit can be implemented on a planar circuit board. We can distinguish between two chemical compounds with the same molecular formula but different structures using graphs. We can determine whether two computers are connected by a communications link using graph models of computer networks. Graphs with weights assigned to their edges can be used to solve problems such as finding the shortest path between two cities in a transportation network. We can also use graphs to schedule exams and assign channels to television stations. This chapter will introduce the basic concepts of graph theory and present many different graph models. To solve the wide variety of problems that can be studied using graphs, we will introduce many different graph algorithms. We will also study the complexity of these algorithms.

## 10.1 Graphs and Graph Models

We begin with the definition of a graph.

**DEFINITION 1** 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.

***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 book we will usually consider only finite graphs.

Now suppose that a network is made up of data centers and communication links between computers. We can represent the location of each data center by a point and each communications link by a line segment, as shown in Figure 1.

This computer network can be modeled using a graph in which the vertices of the graph represent the data centers and the edges represent communication links. In general, we visualize

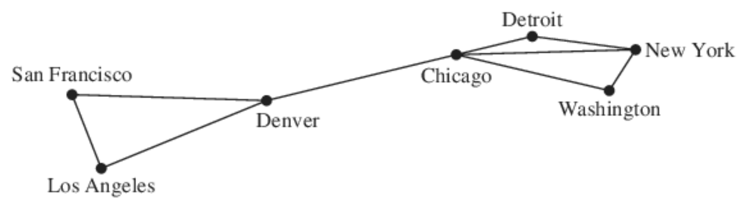



FIGURE 1 **A Computer Network.**

graphs by using points to represent vertices and line segments, possibly curved, to represent edges, where the endpoints of a line segment representing an edge are the points representing the endpoints of the edge. When we draw a graph, we generally try to draw edges so that they do not cross. However, this is not necessary because any depiction using points to represent vertices and any form of connection between vertices can be used. Indeed, there are some graphs that cannot be drawn in the plane without edges crossing (see Section 10.7). The key point is that the way we draw a graph is arbitrary, as long as the correct connections between vertices are depicted.

Note that each edge of the graph representing this computer network connects two different vertices. That is, no edge connects a vertex to itself. Furthermore, no two different edges connect the same pair of 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**. Note that in a simple graph, each edge is associated to an unordered pair of vertices, and no other edge is associated to this same edge. Consequently, when there is an edge of a simple graph associated to $\{u, v\}$, we can also say, without possible confusion, that $\{u, v\}$ is an edge of the graph.

A computer network may contain multiple links between data centers, as shown in Figure 2. To model such networks we need graphs that have more than one edge connecting the same pair of vertices. Graphs that may have **multiple edges** connecting the same vertices are called **multigraphs**. When there are $m$ different edges associated to the same unordered pair of vertices $\{u, v\}$, we also say that $\{u, v\}$ is an edge of multiplicity $m$. That is, we can think of this set of edges as $m$ different copies of an edge $\{u, v\}$.

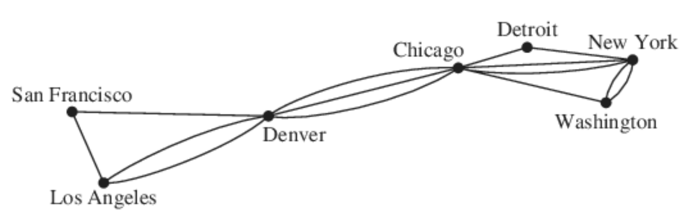



FIGURE 2 **A Computer Network with Multiple Links between Data Centers.**

Sometimes a communications link connects a data center with itself, perhaps a feedback loop for diagnostic purposes. Such a network is illustrated in Figure 3. To model this network we

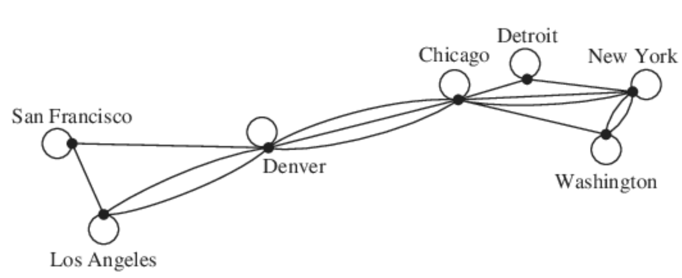



FIGURE 3 **A Computer Network with Diagnostic Links.**

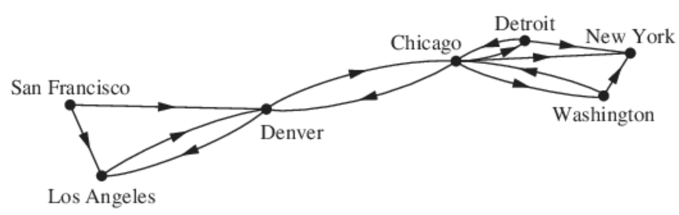



FIGURE 4 **A Communications Network with One-Way Communications Links.**

need to include edges that connect a vertex to itself. Such edges are called **loops**, and sometimes we may even have more than one loop at a vertex. Graphs that may include loops, and possibly multiple edges connecting the same pair of vertices or a vertex to itself, are sometimes called **pseudographs**.

So far the graphs we have introduced are **undirected graphs**. Their edges are also said to be **undirected**. However, to construct a graph model, we may find it necessary to assign directions to the edges of a graph. For example, in a computer network, some links may operate in only one direction (such links are called single duplex lines). This may be the case if there is a large amount of traffic sent to some data centers, with little or no traffic going in the opposite direction. Such a network is shown in Figure 4.

To model such a computer network we use a directed graph. Each edge of a directed graph is associated to an ordered pair. The definition of directed graph we give here is more general than the one we used in Chapter 9, where we used directed graphs to represent relations.

**DEFINITION 2** 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 $v$.

When we depict a directed graph with a line drawing, we use an arrow pointing from $u$ to $v$ to indicate the direction of an edge that starts at $u$ and ends at $v$. A directed graph may contain loops and it may contain multiple directed edges that start and end at the same vertices. A directed graph may also contain directed edges that connect vertices $u$ and $v$ in both directions; that is, when a digraph contains an edge from $u$ to $v$, it may also contain one or more edges from $v$ to $u$. Note that we obtain a directed graph when we assign a direction to each edge in an undirected graph. When a directed graph has no loops and has no multiple directed edges, it is called a **simple directed graph**. Because a simple directed graph has at most one edge associated to each ordered pair of vertices $(u, v)$, we call $(u, v)$ an edge if there is an edge associated to it in the graph.

In some computer networks, multiple communication links between two data centers may be present, as illustrated in Figure 5. 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**. When there are $m$ directed edges, each associated to an ordered pair of vertices $(u, v)$, we say that $(u, v)$ is an edge of **multiplicity** $m$.

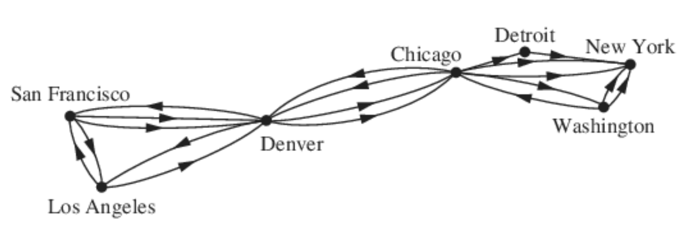



FIGURE 5 **A Computer Network with Multiple One-Way Links.**

**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 |

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**. For example, a mixed graph might be used to model a computer network containing links that operate in both directions and other links that operate only in one direction.

This terminology for the various types of graphs is summarized in Table 1. We will sometimes use the term **graph** as a general term to describe graphs with directed or undirected edges (or both), with or without loops, and with or without multiple edges. At other times, when the context is clear, we will use the term graph to refer only to undirected graphs.

Links

Because of the relatively modern interest in graph theory, and because it has applications to a wide variety of disciplines, many different terminologies of graph theory have been introduced. The reader should determine how such terms are being used whenever they are encountered. The terminology used by mathematicians to describe graphs has been increasingly standardized, but the terminology used to discuss graphs when they are used in other disciplines is still quite varied. Although the terminology used to describe graphs may vary, three key questions can help us understand the structure of a graph:

- Are the edges of the graph undirected or directed (or both)?
- 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?
- Are loops present?

Answering such questions helps us understand graphs. It is less important to remember the particular terminology used.

## Graph Models

Links

Can you find a subject to which graph theory has not been applied?

Graphs are used in a wide variety of models. We began this section by describing how to construct graph models of communications networks linking data centers. We will complete this section by describing some diverse graph models for some interesting applications. We will return to many of these applications later in this chapter and in Chapter 11. We will introduce additional graph models in subsequent sections of this and later chapters. Also, recall that directed graph models for some applications were introduced in Chapter 9. When we build a graph model, we need to make sure that we have correctly answered the three key questions we posed about the structure of a graph.

SOCIAL NETWORKS Graphs are extensively used to model social structures based on different kinds of relationships between people or groups of people. These social structures, and the graphs that represent them, are known as **social networks**. In these graph models, individuals or organizations are represented by vertices; relationships between individuals or organizations are represented by edges. The study of social networks is an extremely active multidisciplinary area, and many different types of relationships between people have been studied using them.

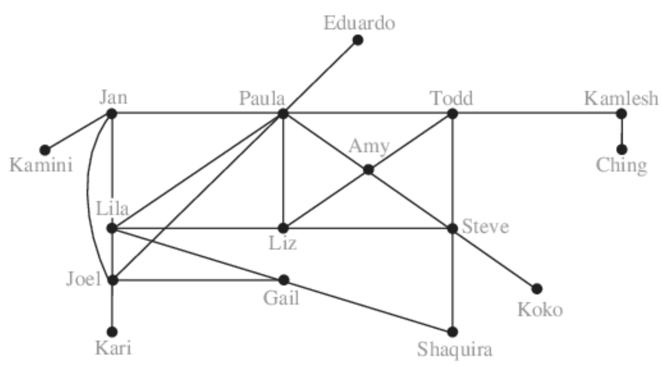



**FIGURE 6** **An Acquaintanceship Graph.**

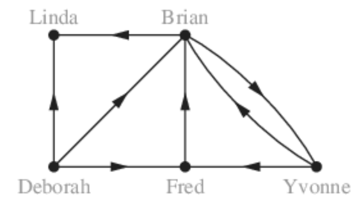



**FIGURE 7** **An Influence Graph.**

We will introduce some of the most commonly studied social networks here. More information about social networks can be found in [Ne10] and [EaKl10].

**EXAMPLE 1** **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 in the virtual world via a social networking site such as Facebook). Each person in a particular group of people is represented by a vertex. An undirected edge is used to connect two people when these people know each other, when we are concerned only with acquaintanceship, or whether they are friends. No multiple edges and usually no loops are used. (If we want to include the notion of self-knowledge, we would include loops.) A small acquaintanceship graph is shown in Figure 6. The acquaintanceship graph of all people in the world has more than six billion vertices and probably more than one trillion edges! We will discuss this graph further in Section 10.4. ◀

**EXAMPLE 2** **Influence Graphs** In studies of group behavior it is observed that certain people can influence the thinking of others. A directed graph called an **influence graph** can be used to model this behavior. Each person of the group is represented by a vertex. There is a directed edge from vertex $a$ to vertex $b$ when the person represented by vertex $a$ can influence the person represented by vertex $b$. This graph does not contain loops and it does not contain multiple directed edges. An example of an influence graph for members of a group is shown in Figure 7. In the group modeled by this influence graph, Deborah cannot be influenced, but she can influence Brian, Fred, and Linda. Also, Yvonne and Brian can influence each other. ◀

**EXAMPLE 3** **Collaboration Graphs** A **collaboration graph** is used to model social networks where two people are related by working together in a particular way. Collaboration graphs are simple graphs, as edges in these graphs are undirected and there are no multiple edges or loops. Vertices in these graphs represent people; two people are connected by an undirected edge when the people have collaborated. There are no loops nor multiple edges in these graphs. The **Hollywood 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 1.5 million vertices (as of early 2011). We will discuss some aspects of the Hollywood graph later in Section 10.4.

In an **academic collaboration graph**, vertices represent people (perhaps restricted to members of a certain academic community), 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, and these numbers have grown considerably since then. We will have more to say about this graph in Section 10.4. Collaboration graphs have also been used in sports, where two professional athletes are considered to have collaborated if they have ever played on the same team during a regular season of their sport. ◀

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. We have already modeled a data network in the first part of this section.

EXAMPLE 4 **Call Graphs** Graphs can be used to model telephone calls made in a network, such as a long-distance telephone network. In particular, a directed multigraph can be used to model calls where each telephone number is represented by a vertex and each telephone call is represented by a directed edge. The edge representing a call starts at the telephone number from which the call was made and ends at the telephone number to which the call was made. We need directed edges because the direction in which the call is made matters. We need multiple directed edges because we want to represent each call made from a particular telephone number to a second number.

Links

A small telephone call graph is displayed in Figure 8(a), representing seven telephone numbers. This graph shows, for instance, that three calls have been made from 732-555-1234 to 732-555-9876 and two in the other direction, but no calls have been made from 732-555-4444 to any of the other six numbers except 732-555-0011. When we care only whether there has been a call connecting two telephone numbers, we use an undirected graph with an edge connecting telephone numbers when there has been a call between these numbers. This version of the call graph is displayed in Figure 8(b).

Call graphs that model actual calling activities can be huge. For example, one call graph studied at AT&T, which models calls during 20 days, has about 290 million vertices and 4 billion edges. We will discuss call graphs further in Section 10.4. ◀

INFORMATION NETWORKS Graphs can be used to model various networks that link particular types of information. Here, we will describe how to model the World Wide Web using a graph. We will also describe how to use a graph to model the citations in different types of documents.

EXAMPLE 5 **The Web Graph** The World Wide Web 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 $b$. Because new Web pages are created and others removed somewhere on the Web almost every second, the Web graph changes on an almost continual basis. Many people are studying the properties of the Web graph to better understand the nature of the Web. We will return to Web graphs in Section 10.4, and in Chapter 11 we will explain how the Web graph is used by the Web crawlers that search engines use to create indexes of Web pages. ◀

Links

EXAMPLE 6 **Citation Graphs** Graphs can be used to represent citations in different types of documents, including academic papers, patents, and legal opinions. In such graphs, each document is represented by a vertex, and there is an edge from one document to a second document if the

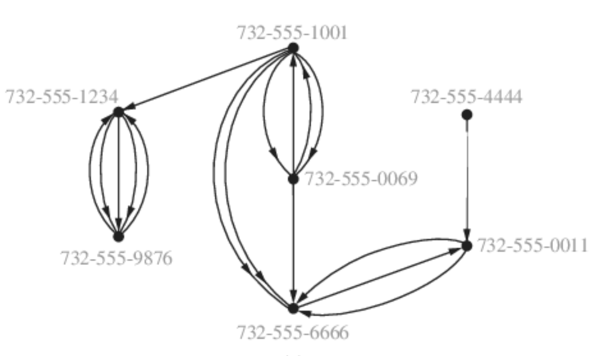



**FIGURE 8 A Call Graph.**

first document cites the second in its citation list. (In an academic paper, the citation list is the bibliography, or list of references; in a patent it is the list of previous patents that are cited; and in a legal opinion it is the list of previous opinions cited.) A citation graph is a directed graph without loops or multiple edges. ◀

SOFTWARE DESIGN APPLICATIONS Graph models are useful tools in the design of software. We will briefly describe two of these models here.

EXAMPLE 7 **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. A **module dependency graph** provides a useful tool for understanding how different modules of a program interact. In a program dependency graph, each module is represented by a vertex. There is a directed edge from a module to a second module if the second module depends on the first. An example of a program dependency graph for a web browser is shown in Figure 9. ◀

EXAMPLE 8 **Precedence Graphs and Concurrent Processing** Computer programs can be executed more rapidly by executing certain statements concurrently. It is important not to execute a statement that requires results of statements not yet executed. The dependence of statements on previous statements can be represented by a directed graph. Each statement is represented by a vertex, and there is an edge from one statement to a second statement if the second statement cannot be executed before the first statement. This resulting graph is called a **precedence graph**. A computer program and its graph are displayed in Figure 10. For instance, the graph shows that statement $S_5$ cannot be executed before statements $S_1$, $S_2$, and $S_4$ are executed. ◀

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

EXAMPLE 9 **Airline Routes** We can model airline networks by representing each airport by a vertex. In particular, we can model all the flights by a particular airline each day using a directed edge to represent each flight, going from the vertex representing the departure airport to the vertex representing the destination airport. The resulting graph will generally be a directed multigraph, as there may be multiple flights from one airport to some other airport during the same day. ◀

EXAMPLE 10 **Road Networks** Graphs can be used to model road networks. In such models, vertices represent intersections and edges represent roads. When all roads are two-way and there is at most one road connecting two intersections, we can use a simple undirected graph to model the road network. However, we will often want to model road networks when some roads are one-way and when there may be more than one road between two intersections. To build such models, we use undirected edges to represent two-way roads and we use directed edges to represent

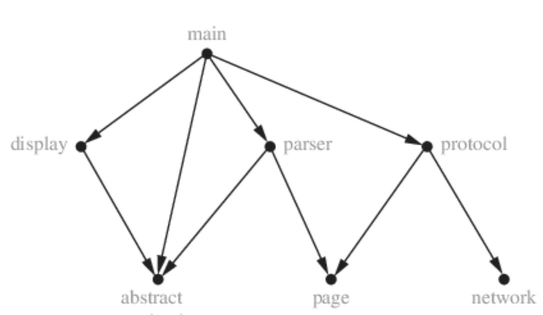



FIGURE 9 **A Module Dependency Graph.**

$S_1$ $a := 0$

$S_2$ $b := 1$

$S_3$ $c := a + 1$

$S_4$ $d := b + a$

$S_5$ $e := d + 1$

$S_6$ $e := c + d$

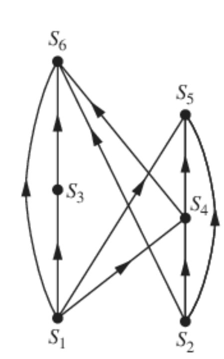



FIGURE 10 **A Precedence Graph.**

one-way roads. Multiple undirected edges represent multiple two-way roads connecting the same two intersections. Multiple directed edges represent multiple one-way roads that start at one intersection and end at a second intersection. Loops represent loop roads. Mixed graphs are needed to model road networks that include both one-way and two-way roads. ◀

BIOLOGICAL NETWORKS Many aspects of the biological sciences can be modeled using graphs.

EXAMPLE 11 **Niche Overlap Graphs in Ecology** Graphs are used in many models involving the interaction of different species of animals. For instance, the competition between species in an ecosystem can be modeled using a **niche overlap graph**. Each species is represented by a vertex. An undirected edge connects two vertices if the two species represented by these vertices compete (that is, some of the food resources they use are the same). A niche overlap graph is a simple graph because no loops or multiple edges are needed in this model. The graph in Figure 11 models the ecosystem of a forest. We see from this graph that squirrels and raccoons compete but that crows and shrews do not. ◀

Links

Extra Examples

EXAMPLE 12 **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. Because protein interactions are crucial for most biological functions, many scientists work on discovering new proteins and understanding interactions between proteins. Protein interactions within a cell can be modeled using a **protein interaction graph** (also called a **protein–protein interaction network**), an undirected graph in which each protein is represented by a vertex, with an edge connecting the vertices representing each pair of proteins that interact. It is a challenging problem to determine genuine protein interactions in a cell, as experiments often produce false positives, which conclude that two proteins interact when they really do not. Protein interaction graphs can be used to deduce important biological information, such as by identifying the most important proteins for various functions and the functionality of newly discovered proteins.

Because there are thousands of different proteins in a typical cell, the protein interaction graph of a cell is extremely large and complex. For example, yeast cells have more than 6,000 proteins, and more than 80,000 interactions between them are known, and human cells have more than 100,000 proteins, with perhaps as many as 1,000,000 interactions between them. Additional vertices and edges are added to a protein interaction graph when new proteins and interactions between proteins are discovered. Because of the complexity of protein interaction graphs, they are often split into smaller graphs called modules that represent groups of proteins that are involved in a particular function of a cell. Figure 12 illustrates a module of the protein interaction graph described in [Bo04], comprising the complex of proteins that degrade RNA in human cells. To learn more about protein interaction graphs, see [Bo04], [Ne10], and [Hu07]. ◀

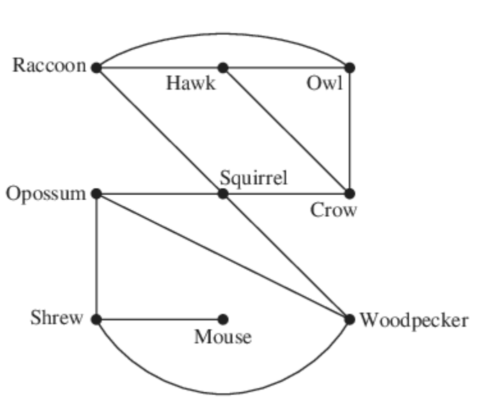



FIGURE 11 **A Niche Overlap Graph.**

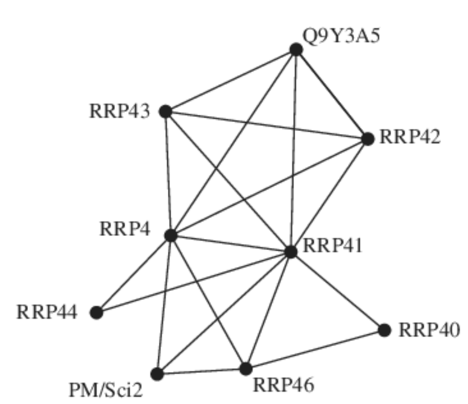



FIGURE 12 **A Module of a Protein Interaction Graph.**

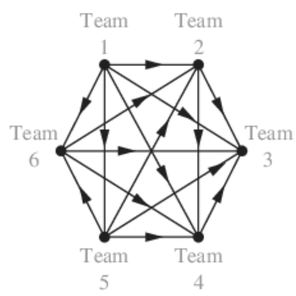



**FIGURE 13 A Graph Model of a Round-Robin Tournament.**

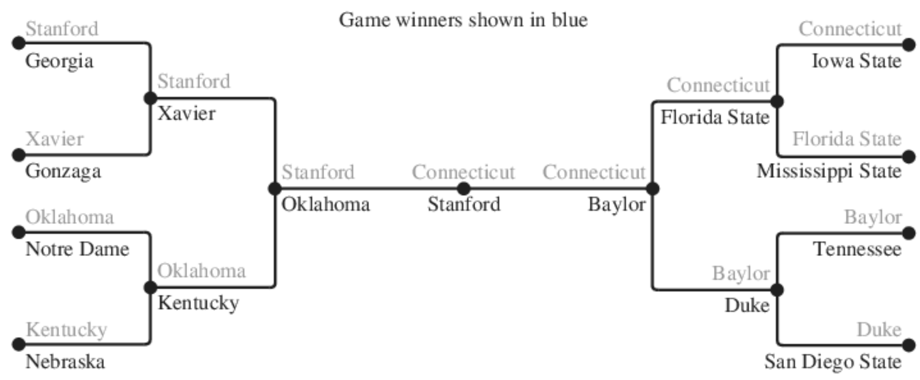



**FIGURE 14 A Single-Elimination Tournament.**

TOURNAMENTS We now give some examples that show how graphs can also be used to model different kinds of tournaments.

EXAMPLE 13 **Round-Robin Tournaments** A tournament where each team plays every other team exactly once and no ties are allowed is called a **round-robin tournament**. Such tournaments can be modeled using directed graphs where each team is represented by a vertex. Note that $(a, b)$ is an edge if team $a$ beats team $b$. This graph is a simple directed graph, containing no loops or multiple directed edges (because no two teams play each other more than once). Such a directed graph model is presented in Figure 13. We see that Team 1 is undefeated in this tournament, and Team 3 is winless. ◀

EXAMPLE 14 **Single-Elimination Tournaments** A tournament where each contestant is eliminated after one loss is called a **single-elimination tournament**. Single-elimination tournaments are often used in sports, including tennis championships and the yearly NCAA basketball championship. We can model such a tournament using a vertex to represent each game and a directed edge to connect a game to the next game the winner of this game played in. The graph in Figure 14 represents the games played by the final 16 teams in the 2010 NCAA women's basketball tournament. ◀

## Exercises

**1.** Draw graph models, stating the type of graph (from Table 1) used, to represent airline routes where every day there are four flights from Boston to Newark, two flights from Newark to Boston, three flights from Newark to Miami, two flights from Miami to Newark, one flight from Newark to Detroit, two flights from Detroit to Newark, three flights from Newark to Washington, two flights from Washington to Newark, and one flight from Washington to Miami, with

**a)** an edge between vertices representing cities that have a flight between them (in either direction).

**b)** an edge between vertices representing cities for each flight that operates between them (in either direction).

**c)** an edge between vertices representing cities for each flight that operates between them (in either direction), plus a loop for a special sightseeing trip that takes off and lands in Miami.

**d)** an edge from a vertex representing a city where a flight starts to the vertex representing the city where it ends.

**e)** an edge for each flight from a vertex representing a city where the flight begins to the vertex representing the city where the flight ends.

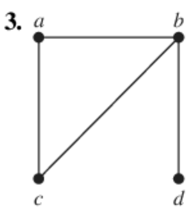
**2.** What kind of graph (from Table 1) can be used to model a highway system between major cities where

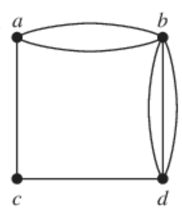
**a)** there is an edge between the vertices representing cities if there is an interstate highway between them?

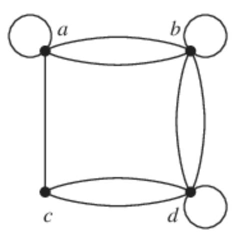
**b)** there is an edge between the vertices representing cities for each interstate highway between them?

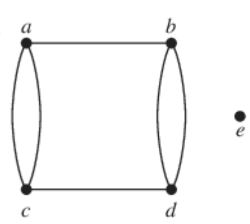
**c)** there is an edge between the vertices representing cities for each interstate highway between them, and there is a loop at the vertex representing a city if there is an interstate highway that circles this city?

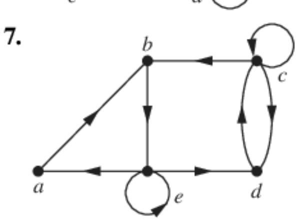
For Exercises 3–9, determine whether the graph shown has directed or undirected edges, whether it has multiple edges, and whether it has one or more loops. Use your answers to determine the type of graph in Table 1 this graph is.

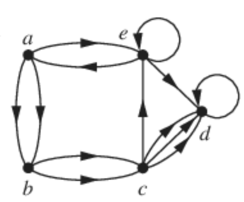
**3.**


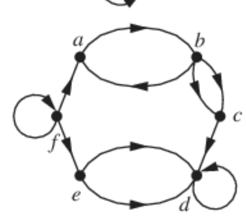

**4.**



**5.**



**6.**



**7.**



**8.**



**9.**



**10.** For each undirected graph in Exercises 3–9 that is not simple, find a set of edges to remove to make it simple.

**11.** Let $G$ be a simple graph. Show that the relation $R$ on the set of vertices of $G$ such that $uRv$ if and only if there is an edge associated to $\{u, v\}$ is a symmetric, irreflexive relation on $G$.

**12.** Let $G$ be an undirected graph with a loop at every vertex. Show that the relation $R$ on the set of vertices of $G$ such that $uRv$ if and only if there is an edge associated to $\{u, v\}$ is a symmetric, reflexive relation on $G$.

**13.** The **intersection graph** of a collection of sets $A_1, A_2, \ldots, A_n$ is the graph that has a vertex for each of these sets and has an edge connecting the vertices representing two sets if these sets have a nonempty intersection. Construct the intersection graph of these collections of sets.

**a)** $A_1 = \{0, 2, 4, 6, 8\}$, $A_2 = \{0, 1, 2, 3, 4\}$,
$A_3 = \{1, 3, 5, 7, 9\}$, $A_4 = \{5, 6, 7, 8, 9\}$,
$A_5 = \{0, 1, 8, 9\}$

**b)** $A_1 = \{\ldots, -4, -3, -2, -1, 0\}$,
$A_2 = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$,
$A_3 = \{\ldots, -6, -4, -2, 0, 2, 4, 6, \ldots\}$,
$A_4 = \{\ldots, -5, -3, -1, 1, 3, 5, \ldots\}$,
$A_5 = \{\ldots, -6, -3, 0, 3, 6, \ldots\}$

**c)** $A_1 = \{x \mid x < 0\}$,
$A_2 = \{x \mid -1 < x < 0\}$,
$A_3 = \{x \mid 0 < x < 1\}$,
$A_4 = \{x \mid -1 < x < 1\}$,
$A_5 = \{x \mid x > -1\}$,
$A_6 = \mathbf{R}$

**14.** Use the niche overlap graph in Figure 11 to determine the species that compete with hawks.

**15.** Construct a niche overlap graph for six species of birds, where the hermit thrush competes with the robin and with the blue jay, the robin also competes with the mockingbird, the mockingbird also competes with the blue jay, and the nuthatch competes with the hairy woodpecker.

**16.** Draw the acquaintanceship graph that represents that Tom and Patricia, Tom and Hope, Tom and Sandy, Tom and Amy, Tom and Marika, Jeff and Patricia, Jeff and Mary, Patricia and Hope, Amy and Hope, and Amy and Marika know each other, but none of the other pairs of people listed know each other.

**17.** We can use a graph to represent whether two people were alive at the same time. Draw such a graph to represent whether each pair of the mathematicians and computer scientists with biographies in the first five chapters of this book who died before 1900 were contemporaneous. (Assume two people lived at the same time if they were alive during the same year.)

**18.** Who can influence Fred and whom can Fred influence in the influence graph in Example 2?

**19.** Construct an influence graph for the board members of a company if the President can influence the Director of Research and Development, the Director of Marketing, and the Director of Operations; the Director of Research and Development can influence the Director of Operations; the Director of Marketing can influence the Director of Operations; and no one can influence, or be influenced by, the Chief Financial Officer.

**20.** Which other teams did Team 4 beat and which teams beat Team 4 in the round-robin tournament represented by the graph in Figure 13?

**21.** In a round-robin tournament the Tigers beat the Blue Jays, the Tigers beat the Cardinals, the Tigers beat the Orioles, the Blue Jays beat the Cardinals, the Blue Jays beat the Orioles, and the Cardinals beat the Orioles. Model this outcome with a directed graph.

**22.** Construct the call graph for a set of seven telephone numbers 555-0011, 555-1221, 555-1333, 555-8888, 555-2222, 555-0091, and 555-1200 if there were three calls from 555-0011 to 555-8888 and two calls from 555-8888 to 555-0011, two calls from 555-2222 to 555-0091, two calls from 555-1221 to each of the other numbers, and one call from 555-1333 to each of 555-0011, 555-1221, and 555-1200.

**23.** Explain how the two telephone call graphs for calls made during the month of January and calls made during the month of February can be used to determine the new telephone numbers of people who have changed their telephone numbers.

24. a) Explain how graphs can be used to model electronic mail messages in a network. Should the edges be directed or undirected? Should multiple edges be allowed? Should loops be allowed?
b) Describe a graph that models the electronic mail sent in a network in a particular week.
25. How can a graph that models e-mail messages sent in a network be used to find people who have recently changed their primary e-mail address?
26. How can a graph that models e-mail messages sent in a network be used to find electronic mail mailing lists used to send the same message to many different e-mail addresses?
27. Describe a graph model that represents whether each person at a party knows the name of each other person at the party. Should the edges be directed or undirected? Should multiple edges be allowed? Should loops be allowed?
28. Describe a graph model that represents a subway system in a large city. Should edges be directed or undirected? Should multiple edges be allowed? Should loops be allowed?
29. For each course at a university, there may be one or more other courses that are its prerequisites. How can a graph be used to model these courses and which courses are prerequisites for which courses? Should edges be directed or undirected? Looking at the graph model, how can we find courses that do not have any prerequisites and how can we find courses that are not the prerequisite for any other courses?
30. Describe a graph model that represents the positive recommendations of movie critics, using vertices to represent both these critics and all movies that are currently being shown.
31. Describe a graph model that represents traditional marriages between men and women. Does this graph have any special properties?
32. Which statements must be executed before $S_6$ is executed in the program in Example 8? (Use the precedence graph in Figure 10.)
33. Construct a precedence graph for the following program:

$S_1$: $x := 0$
$S_2$: $x := x + 1$
$S_3$: $y := 2$
$S_4$: $z := y$
$S_5$: $x := x + 2$
$S_6$: $y := x + z$
$S_7$: $z := 4$

34. Describe a discrete structure based on a graph that can be used to model airline routes and their flight times. [*Hint:* Add structure to a directed graph.]
35. Describe a discrete structure based on a graph that can be used to model relationships between pairs of individuals in a group, where each individual may either like, dislike, or be neutral about another individual, and the reverse relationship may be different. [*Hint:* Add structure to a directed graph. Treat separately the edges in opposite directions between vertices representing two individuals.]
36. Describe a graph model that can be used to represent all forms of electronic communication between two people in a single graph. What kind of graph is needed?

# 10.2 Graph Terminology and Special Types of Graphs

## Introduction

Links

We introduce some of the basic vocabulary of graph theory in this section. We will use this vocabulary later in this chapter when we solve many different types of problems. One such problem involves determining whether a graph can be drawn in the plane so that no two of its edges cross. Another example is deciding whether there is a one-to-one correspondence between the vertices of two graphs that produces a one-to-one correspondence between the edges of the graphs. We will also introduce several important families of graphs often used as examples and in models. Several important applications will be described where these special types of graphs arise.

## Basic Terminology

First, we give some terminology that describes the vertices and edges of undirected graphs.

**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 $v$.

We will also find useful terminology describing the set of vertices adjacent to a particular vertex of a graph.

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

To keep track of how many edges are incident to a vertex, we make the following definition.

**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 $\deg(v)$.

**EXAMPLE 1** What are the degrees and what are the neighborhoods of the vertices in the graphs $G$ and $H$ displayed in Figure 1?

*Solution:* In $G$, $\deg(a) = 2$, $\deg(b) = \deg(c) = \deg(f) = 4$, $\deg(d) = 1$, $\deg(e) = 3$, and $\deg(g) = 0$. The neighborhoods of these vertices are $N(a) = \{b, f\}$, $N(b) = \{a, c, e, f\}$, $N(c) = \{b, d, e, f\}$, $N(d) = \{c\}$, $N(e) = \{b, c, f\}$, $N(f) = \{a, b, c, e\}$, and $N(g) = \emptyset$. In $H$, $\deg(a) = 4$, $\deg(b) = \deg(e) = 6$, $\deg(c) = 1$, and $\deg(d) = 5$. The neighborhoods of these vertices are $N(a) = \{b, d, e\}$, $N(b) = \{a, b, c, d, e\}$, $N(c) = \{b\}$, $N(d) = \{a, b, e\}$, and $N(e) = \{a, b, d\}$. ◀

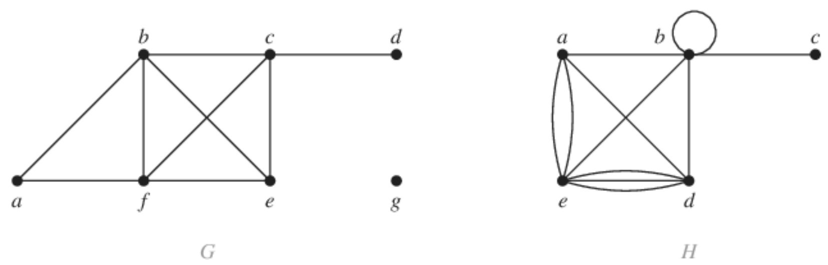



**FIGURE 1 The Undirected Graphs $G$ and $H$.**

A vertex of degree zero is called **isolated**. It follows that an isolated vertex is not adjacent to any vertex. Vertex $g$ in graph $G$ in Example 1 is isolated. A vertex is **pendant** if and only if it has degree one. Consequently, a pendant vertex is adjacent to exactly one other vertex. Vertex $d$ in graph $G$ in Example 1 is pendant.

Examining the degrees of vertices in a graph model can provide useful information about the model, as Example 2 shows.

**EXAMPLE 2** What does the degree of a vertex in a niche overlap graph (introduced in Example 11 in Section 10.1) represent? Which vertices in this graph are pendant and which are isolated? Use the niche overlap graph shown in Figure 11 of Section 10.1 to interpret your answers.

*Solution:* There is an edge between two vertices in a niche overlap graph if and only if the two species represented by these vertices compete. Hence, the degree of a vertex in a niche overlap graph is the number of species in the ecosystem that compete with the species represented by this vertex. A vertex is pendant if the species competes with exactly one other species in the ecosystem. Finally, the vertex representing a species is isolated if this species does not compete with any other species in the ecosystem.

For instance, the degree of the vertex representing the squirrel in the niche overlap graph in Figure 11 in Section 10.1 is four, because the squirrel competes with four other species: the crow, the opossum, the raccoon, and the woodpecker. In this niche overlap graph, the mouse is the only species represented by a pendant vertex, because the mouse competes only with the shrew and all other species compete with at least two other species. There are no isolated vertices in the graph in this niche overlap graph because every species in this ecosystem competes with at least one other species. ◀

What do we get when we add the degrees of all the vertices of a graph $G = (V, E)$? Each edge contributes two to the sum of the degrees of the vertices because an edge is incident with exactly two (possibly equal) vertices. This means that the sum of the degrees of the vertices is twice the number of edges. We have the result in Theorem 1, which is sometimes called the handshaking theorem (and is also often known as the handshaking lemma), because of the analogy between an edge having two endpoints and a handshake involving two hands.

**THEOREM 1** **THE HANDSHAKING THEOREM** Let $G = (V, E)$ be an undirected graph with $m$ edges. Then

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

(Note that this applies even if multiple edges and loops are present.)

**EXAMPLE 3** How many edges are there in a graph with 10 vertices each of degree six?

*Solution:* Because the sum of the degrees of the vertices is $6 \cdot 10 = 60$, it follows that $2m = 60$ where $m$ is the number of edges. Therefore, $m = 30$. ◀

Theorem 1 shows that the sum of the degrees of the vertices of an undirected graph is even. This simple fact has many consequences, one of which is given as Theorem 2.

**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. Then

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

Because $\deg(v)$ is even for $v \in V_1$, the first term in the right-hand side of the last equality is even. Furthermore, the sum of the two terms on the right-hand side of the last equality is even, because this sum is $2m$. Hence, the second term in the sum is also even. Because all the terms in this sum are odd, there must be an even number of such terms. Thus, there are an even number of vertices of odd degree. ◁

Terminology for graphs with directed edges reflects the fact that edges in directed graphs have directions.

**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.

Because the edges in graphs with directed edges are ordered pairs, the definition of the degree of a vertex can be refined to reflect the number of edges with this vertex as the initial vertex and as the terminal vertex.

**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.)

**EXAMPLE 4** Find the in-degree and out-degree of each vertex in the graph $G$ with directed edges shown in Figure 2.

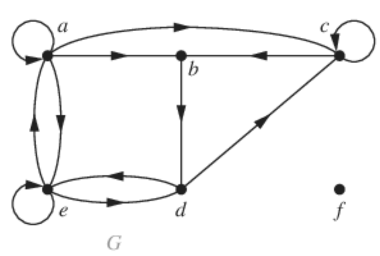



**FIGURE 2 The Directed Graph *G*.**

*Solution:* The in-degrees in $G$ are $\deg^-(a) = 2$, $\deg^-(b) = 2$, $\deg^-(c) = 3$, $\deg^-(d) = 2$, $\deg^-(e) = 3$, and $\deg^-(f) = 0$. The out-degrees are $\deg^+(a) = 4$, $\deg^+(b) = 1$, $\deg^+(c) = 2$, $\deg^+(d) = 2$, $\deg^+(e) = 3$, and $\deg^+(f) = 0$. ◀

Because each edge has an initial vertex and a terminal vertex, the sum of the in-degrees and the sum of the out-degrees of all vertices in a graph with directed edges are the same. Both of these sums are the number of edges in the graph. This result is stated as Theorem 3.

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

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

There are many properties of a graph with directed edges that do not depend on the direction of its edges. Consequently, it is often useful to ignore these directions. The undirected graph that results from ignoring directions of edges is called the **underlying undirected graph**. A graph with directed edges and its underlying undirected graph have the same number of edges.

## Some Special Simple Graphs

We will now introduce several classes of simple graphs. These graphs are often used as examples and arise in many applications.

**EXAMPLE 5** **Complete Graphs** A **complete graph on $n$ vertices**, denoted by $K_n$, is a 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 displayed in Figure 3. A simple graph for which there is at least one pair of distinct vertex not connected by an edge is called **noncomplete**. ◀

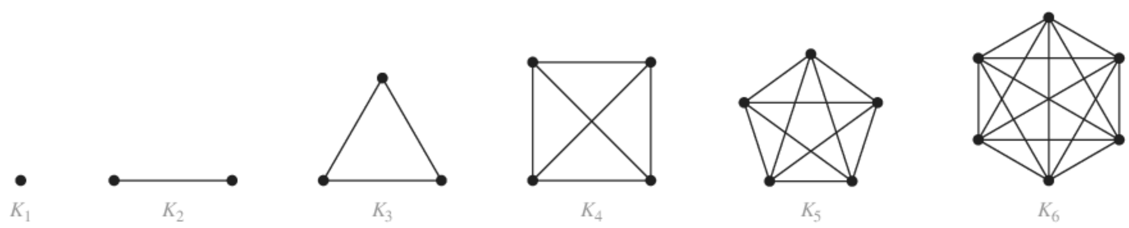



**FIGURE 3** **The Graphs $K_n$ for $1 \le n \le 6$.**

**EXAMPLE 6** **Cycles** A **cycle $C_n$**, $n \ge 3$, 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 displayed in Figure 4. ◀

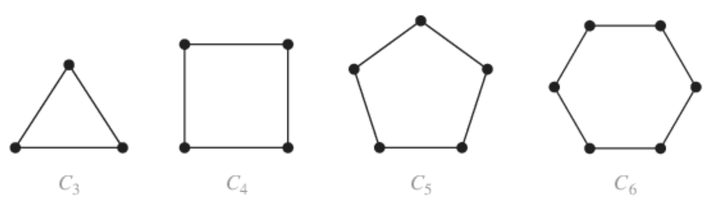



**FIGURE 4** **The Cycles $C_3$, $C_4$, $C_5$, and $C_6$.**

**EXAMPLE 7** **Wheels** We obtain a **wheel $W_n$** when we add an additional vertex to a cycle $C_n$, for $n \ge 3$, 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 displayed in Figure 5. ◀

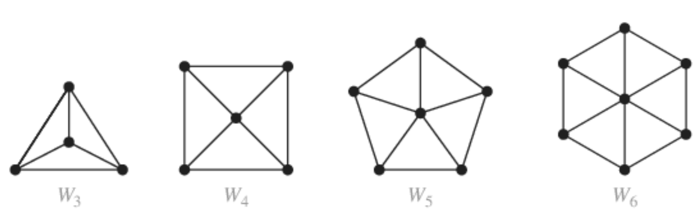



**FIGURE 5** **The Wheels $W_3$, $W_4$, $W_5$, and $W_6$.**

**EXAMPLE 8** ***n*-Cubes** An ***n*-dimensional hypercube**, or ***n*-cube**, denoted by $Q_n$, 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. We display $Q_1$, $Q_2$, and $Q_3$ in Figure 6.

Note that you can construct the $(n + 1)$-cube $Q_{n+1}$ from the $n$-cube $Q_n$ by making two copies of $Q_n$, prefacing the labels on the vertices with a 0 in one copy of $Q_n$ and with a 1 in the other copy of $Q_n$, and adding edges connecting two vertices that have labels differing only in the first bit. In Figure 6, $Q_3$ is constructed from $Q_2$ by drawing two copies of $Q_2$ as the top and bottom faces of $Q_3$, adding 0 at the beginning of the label of each vertex in the bottom face and 1 at the beginning of the label of each vertex in the top face. (Here, by *face* we mean a face of a cube in three-dimensional space. Think of drawing the graph $Q_3$ in three-dimensional space with copies of $Q_2$ as the top and bottom faces of a cube and then drawing the projection of the resulting depiction in the plane.) ◀

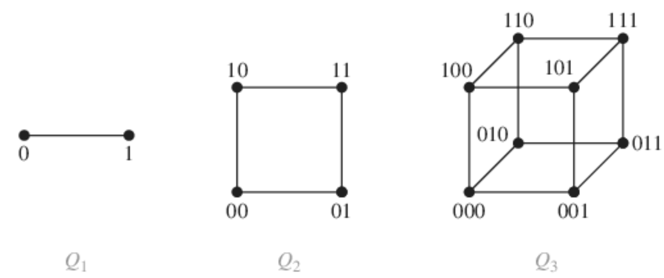



FIGURE 6 **The $n$-cube $Q_n$, $n = 1, 2, 3$.**

## Bipartite Graphs

Sometimes a graph has the property that its vertex set can be divided into two disjoint subsets such that each edge connects a vertex in one of these subsets to a vertex in the other subset. For example, consider the graph representing marriages between men and women in a village, where each person is represented by a vertex and a marriage is represented by an edge. In this graph, each edge connects a vertex in the subset of vertices representing males and a vertex in the subset of vertices representing females. This leads us to Definition 5.

Links

**DEFINITION 6** A simple graph $G$ is called *bipartite* if its vertex set $V$ can be partitioned into two disjoint sets $V_1$ and $V_2$ 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 $G$.

In Example 9 we will show that $C_6$ is bipartite, and in Example 10 we will show that $K_3$ is not bipartite.

**EXAMPLE 9** $C_6$ is bipartite, as shown in Figure 7, because its vertex set can be partitioned into the two sets $V_1 = \{v_1, v_3, v_5\}$ and $V_2 = \{v_2, v_4, v_6\}$, and every edge of $C_6$ connects a vertex in $V_1$ and a vertex in $V_2$. ◀

**EXAMPLE 10** $K_3$ is not bipartite. To verify this, note that if we divide the vertex set of $K_3$ into two disjoint sets, one of the two sets must contain two vertices. If the graph were bipartite, these two vertices could not be connected by an edge, but in $K_3$ each vertex is connected to every other vertex by an edge. ◀

**EXAMPLE 11** Are the graphs $G$ and $H$ displayed in Figure 8 bipartite?

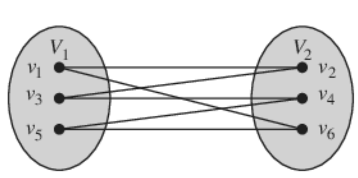



FIGURE 7 **Showing That $C_6$ Is Bipartite.**

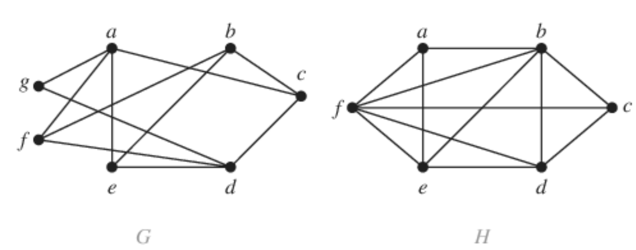



FIGURE 8 **The Undirected Graphs $G$ and $H$.**

*Solution:* Graph $G$ is bipartite because its vertex set is the union of two disjoint sets, $\{a, b, d\}$ and $\{c, e, f, g\}$, and each edge connects a vertex in one of these subsets to a vertex in the other subset. (Note that for $G$ to be bipartite it is not necessary that every vertex in $\{a, b, d\}$ be adjacent to every vertex in $\{c, e, f, g\}$. For instance, $b$ and $g$ are not adjacent.)

Graph $H$ is not bipartite because its vertex set cannot be partitioned into two subsets so that edges do not connect two vertices from the same subset. (The reader should verify this by considering the vertices $a$, $b$, and $f$.) ◀

Theorem 4 provides a useful criterion for determining whether a graph is bipartite.

**THEOREM 4** 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.

*Proof:* First, suppose that $G = (V, E)$ is a bipartite simple graph. Then $V = V_1 \cup V_2$, where $V_1$ and $V_2$ are disjoint sets and every edge in $E$ connects a vertex in $V_1$ and a vertex in $V_2$. If we assign one color to each vertex in $V_1$ and a second color to each vertex in $V_2$, then no two adjacent vertices are assigned the same color.

Now suppose that it is possible to assign colors to the vertices of the graph using just two colors so that no two adjacent vertices are assigned the same color. Let $V_1$ be the set of vertices assigned one color and $V_2$ be the set of vertices assigned the other color. Then, $V_1$ and $V_2$ are disjoint and $V = V_1 \cup V_2$. Furthermore, every edge connects a vertex in $V_1$ and a vertex in $V_2$ because no two adjacent vertices are either both in $V_1$ or both in $V_2$. Consequently, $G$ is bipartite. ◁

We illustrate how Theorem 4 can be used to determine whether a graph is bipartite in Example 12.

**EXAMPLE 12** Use Theorem 4 to determine whether the graphs in Example 11 are bipartite.

*Solution:* We first consider the graph $G$. We will try to assign one of two colors, say red and blue, to each vertex in $G$ so that no edge in $G$ connects a red vertex and a blue vertex. Without loss of generality we begin by arbitrarily assigning red to $a$. Then, we must assign blue to $c$, $e$, $f$, and $g$, because each of these vertices is adjacent to $a$. To avoid having an edge with two blue endpoints, we must assign red to all the vertices adjacent to either $c$, $e$, $f$, or $g$. This means that we must assign red to both $b$ and $d$ (and means that $a$ must be assigned red, which it already has been). We have now assigned colors to all vertices, with $a$, $b$, and $d$ red and $c$, $e$, $f$, and $g$ blue. Checking all edges, we see that every edge connects a red vertex and a blue vertex. Hence, by Theorem 4 the graph $G$ is bipartite.

Next, we will try to assign either red or blue to each vertex in $H$ so that no edge in $H$ connects a red vertex and a blue vertex. Without loss of generality we arbitrarily assign red to $a$. Then, we must assign blue to $b$, $e$, and $f$, because each is adjacent to $a$. But this is not possible because $e$ and $f$ are adjacent, so both cannot be assigned blue. This argument shows that we cannot assign one of two colors to each of the vertices of $H$ so that no adjacent vertices are assigned the same color. It follows by Theorem 4 that $H$ is not bipartite. ◀

Theorem 4 is an example of a result in the part of graph theory known as graph colorings. Graph colorings is an important part of graph theory with important applications. We will study graph colorings further in Section 10.8.

Another useful criterion for determining whether a graph is bipartite is based on the notion of a path, a topic we study in Section 10.4. A graph is bipartite if and only if it is not possible to start at a vertex and return to this vertex by traversing an odd number of distinct edges. We will make this notion more precise when we discuss paths and circuits in graphs in Section 10.4 (see Exercise 63 in that section).

EXAMPLE 13 **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. The complete bipartite graphs $K_{2,3}$, $K_{3,3}$, $K_{3,5}$, and $K_{2,6}$ are displayed in Figure 9. ◀

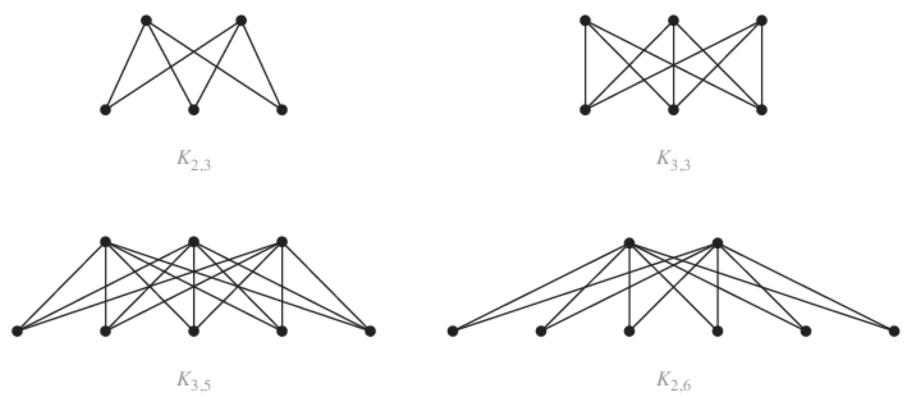



FIGURE 9 **Some Complete Bipartite Graphs.**

## Bipartite Graphs and Matchings

Bipartite graphs can be used to model many types of applications that involve matching the elements of one set to elements of another, as Example 14 illustrates.

EXAMPLE 14 **Job Assignments** Suppose that there are $m$ employees in a group and $n$ different jobs that need to be done, where $m \geq n$. Each employee is trained to do one or more of these $n$ jobs. We would like to assign an employee to each job. To help with this task, we can use a graph to model employee capabilities. We represent each employee by a vertex and each job by a vertex. For each employee, we include an edge from that employee to all jobs that the employee has been trained to do. Note that the vertex set of this graph can be partitioned into two disjoint sets, the set of employees and the set of jobs, and each edge connects an employee to a job. Consequently, this graph is bipartite, where the bipartition is $(E, J)$ where $E$ is the set of employees and $J$ is the set of jobs. We now consider two different scenarios.

First, suppose that a group has four employees: Alvarez, Berkowitz, Chen, and Davis; and suppose that four jobs need to be done to complete Project 1: requirements, architecture, implementation, and testing. Suppose that Alvarez has been trained to do requirements and testing; Berkowitz has been trained to do architecture, implementation, and testing; Chen has been trained to do requirements, architecture, and implementation; and Davis has only been trained to do requirements. We model these employee capabilities using the bipartite graph in Figure 10(a).

Second, suppose that a group has second group also has four employees: Washington, Xuan, Ybarra, and Ziegler; and suppose that the same four jobs need to be done to complete Project 2 as are needed to complete Project 1. Suppose that Washington has been trained to do architecture; Xuan has been trained to do requirements, implementation, and testing; Ybarra has been trained to do architecture; and Ziegler has been trained to do requirements, architecture and testing. We model these employee capabilities using the bipartite graph in Figure 10(b).

To complete Project 1, we must assign an employee to each job so that every job has an employee assigned to it, and so that no employee is assigned more than one job. We can do this by assigning Alvarez to testing, Berkowitz to implementation, Chen to architecture, and Davis to requirements, as shown in Figure 10(a) (where blue lines show this assignment of jobs).

To complete Project 2, we must also assign an employee to each job so that every job has an employee assigned to it and no employee is assigned more than one job. However, this is

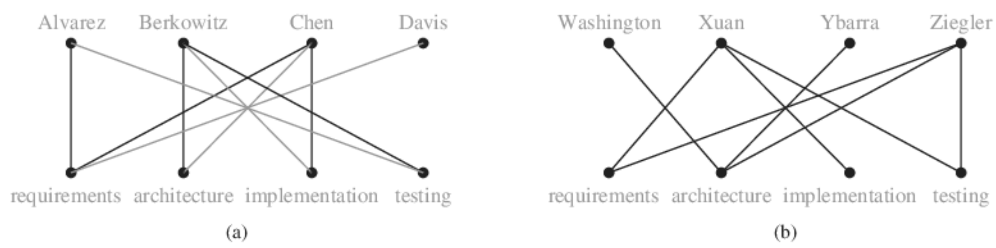



FIGURE 10 **Modeling the Jobs for Which Employees Have Been Trained.**

impossible because there are only two employees, Xuan and Ziegler, who have been trained for at least one of the three jobs of requirements, implementation, and testing. Consequently, there is no way to assign three different employees to these three job so that each job is assigned an employee with the appropriate training. ◀

Finding an assignment of jobs to employees can be thought of as finding a matching in the graph model, where 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. In other words, a matching is a subset of edges such that if $\{s, t\}$ and $\{u, v\}$ are distinct edges of the matching, then $s$, $t$, $u$, and $v$ are distinct. 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. We say that 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 $|M| = |V_1|$. For example, to assign jobs to employees so that the largest number of jobs are assigned employees, we seek a maximum matching in the graph that models employee capabilities. To assign employees to all jobs we seek a complete matching from the set of jobs to the set of employees. In Example 14, we found a complete matching from the set of jobs to the set of employees for Project 1, and this matching is a maximun matching, and we showed that no complete matching exists from the set of jobs to the employees for Project 2.

We now give an example of how matchings can be used to model marriages.

EXAMPLE 15 **Marriages on an Island** Suppose that there are $m$ men and $n$ women on an island. Each person has a list of members of the opposite gender acceptable as a spouse. We construct a bipartite graph $G = (V_1, V_2)$ where $V_1$ is the set of men and $V_2$ is the set of women so that there is an edge between a man and a woman if they find each other acceptable as a spouse. A matching in this graph consists of a set of edges, where each pair of endpoints of an edge is a husband-wife pair. A maximum matching is a largest possible set of married couples, and a complete matching of $V_1$ is a set of married couples where every man is married, but possibly not all women. ◀

NECESSARY AND SUFFICIENT CONDITIONS FOR COMPLETE MATCHINGS We now turn our attention to the question of determining whether a complete matching from $V_1$ to $V_2$ exists when $(V_1, V_2)$ is a bipartition of a bipartite graph $G = (V, E)$. We will introduce a theorem that provides a set of necessary and sufficient conditions for the existence of a complete matching. This theorem was proved by Philip Hall in 1935.

Hall's marriage theorem is an example of a theorem where obvious necessary conditions are sufficient too.

THEOREM 5 HALL'S MARRIAGE THEOREM 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 $V_1$.

*Proof:* We first prove the *only if* part of the theorem. To do so, suppose that there is a complete matching $M$ from $V_1$ to $V_2$. Then, if $A \subseteq V_1$, for every vertex $v \in A$, there is an edge in $M$ connecting $v$ to a vertex in $V_2$. Consequently, there are at least as many vertices in $V_2$ that are neighbors of vertices in $V_1$ as there are vertices in $V_1$. It follows that $|N(A)| \geq |A|$.

To prove the *if* part of the theorem, the more difficult part, we need to show that if $|N(A)| \geq |A|$ for all $A \subseteq V_1$, then there is a complete matching $M$ from $V_1$ to $V_2$. We will use strong induction on $|V_1|$ to prove this.

*Basis step:* If $|V_1| = 1$, then $V_1$ contains a single vertex $v_0$. Because $|N(\{v_0\})| \geq |\{v_0\}| = 1$, there is at least one edge connecting $v_0$ and a vertex $w_0 \in V_2$. Any such edge forms a complete matching from $V_1$ to $V_2$.

*Inductive step:* We first state the inductive hypothesis.

*Inductive hypothesis:* Let $k$ be a positive integer. If $G = (V, E)$ is a bipartite graph with bipartition $(V_1, V_2)$, and $|V_1| = j \leq k$, then there is a complete matching $M$ from $V_1$ to $V_2$ whenever the condition that $|N(A)| \geq |A|$ for all $A \subseteq V_1$ is met.

Now suppose that $H = (W, F)$ is a bipartite graph with bipartition $(W_1, W_2)$ and $|W_1| = k + 1$. We will prove that the inductive holds using a proof by cases, using two case. Case *(i)* applies when for all integers $j$ with $1 \leq j \leq k$, the vertices in every set of $j$ elements from $W_1$ are adjacent to at least $j + 1$ elements of $W_2$. Case *(ii)* applies when for some $j$ with $1 \leq j \leq k$ there is a subset $W_1'$ of $j$ vertices such that there are exactly $j$ neighbors of these vertices in $W_2$. Because either Case *(i)* or Case *(ii)* holds, we need only consider these cases to complete the inductive step.

*Case (i)*: Suppose that for all integers $j$ with $1 \leq j \leq k$, the vertices in every subset of $j$ elements from $W_1$ are adjacent to at least $j + 1$ elements of $W_2$. Then, we select a vertex $v \in W_1$ and an element $w \in N(\{v\})$, which must exist by our assumption that $|N(\{v\}| \geq |\{v\}| = 1$. We delete $v$ and $w$ and all edges incident to them from $H$. This produces a bipartite graph $H'$ with bipartition $(W_1 - \{v\}, W_2 - \{w\})$. Because $|W_1 - \{v\}| = k$, the inductive hypothesis tells us there is a complete matching from $W_1 - \{v\}$ to $W_2 - \{w\}$. Adding the edge from $v$ to $w$ to this complete matching produces a complete matching from $W_1$ to $W_2$.

*Case (ii)*: Suppose that for some $j$ with $1 \leq j \leq k$, there is a subset $W_1'$ of $j$ vertices such that there are exactly $j$ neighbors of these vertices in $W_2$. Let $W_2'$ be the set of these neighbors. Then, by the inductive hypothesis there is a complete matching from $W_1'$ to $W_2'$. Remove these $2j$ vertices from $W_1$ and $W_2$ and all incident edges to produce a bipartite graph $K$ with bipartition $(W_1 - W_1', W_2 - W_2')$.

We will show that the graph $K$ satisfies the condition $|N(A)| \geq |A|$ for all subsets $A$ of $W_1 - W_1'$. If not, there would be a subset of $t$ vertices of $W_1 - W_1'$ where $1 \leq t \leq k + 1 - j$ such that the vertices in this subset have fewer than $t$ vertices of $W_2 - W_2'$ as neighbors. Then, the set of $j + t$ vertices of $W_1$ consisting of these $t$ vertices together with the $j$ vertices we removed from $W_1$ has fewer than $j + t$ neighbors in $W_2$, contradicting the hypothesis that $|N(A)| \geq |A|$ for all $A \subseteq W_1$.

PHILIP HALL (1904–1982) Philip Hall grew up in London, where his mother was a dressmaker. He won a scholarship for board school reserved for needy children, and later a scholarship to King's College of Cambridge University. He received his bachelors degree in 1925. In 1926, unsure of his career goals, he took a civil service exam, but decided to continue his studies at Cambridge after failing.

In 1927 Hall was elected to a fellowship at King's College; soon after, he made his first important discovery in group theory. The results he proved are now known as Hall's theorems. In 1933 he was appointed as a Lecturer at Cambridge, where he remained until 1941. During World War II he worked as a cryptographer at Bletchley Park breaking Italian and Japanese codes. At the end of the war, Hall returned to King's College, and was soon promoted. In 1953 he was appointed to the Sadleirian Chair. His work during the 1950s proved to be extremely influential to the rapid development of group theory during the 1960s.

Hall loved poetry and recited it beautifully in Italian and Japanese, as well as English. He was interested in art, music, and botany. He was quite shy and disliked large groups of people. Hall had an incredibly broad and varied knowledge, and was respected for his integrity, intellectual standards, and judgement. He was beloved by his students.

Hence, by the inductive hypothesis, the graph $K$ has a complete matching. Combining this complete matching with the complete matching from $W_1'$ to $W_2'$, we obtain a complete matching from $W_1$ to $W_2$.

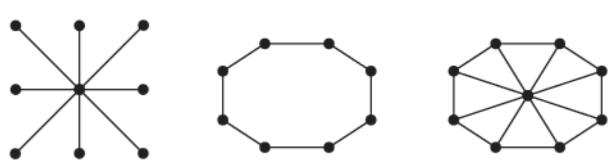
We have shown that in both cases there is a complete matching from $W_1$ to $W_2$. This completes the inductive step and completes the proof. ◁

We have used strong induction to prove Hall's marriage theorem. Although our proof is elegant, it does have some drawbacks. In particular, we cannot construct an algorithm based on this proof that finds a complete matching in a bipartite graph. For a constructive proof that can be used as the basis of an algorithm, see [Gi85].

## Some Applications of Special Types of Graphs

We conclude this section by introducing some additional graph models that involve the special types of graph we have discussed in this section.

EXAMPLE 16 **Local Area Networks** The various computers in a building, such as minicomputers and personal computers, as well as peripheral devices such as printers and plotters, can be connected using a *local area network*. Some of these networks are based on a *star topology*, where all devices are connected to a central control device. A local area network can be represented using a complete bipartite graph $K_{1,n}$, as shown in Figure 11(a). Messages are sent from device to device through the central control device.



(a) (b) (c)

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

Other local area networks are based on a *ring topology*, where each device is connected to exactly two others. Local area networks with a ring topology are modeled using $n$-cycles, $C_n$, as shown in Figure 11(b). Messages are sent from device to device around the cycle until the intended recipient of a message is reached.

Finally, some local area networks use a hybrid of these two topologies. Messages may be sent around the ring, or through a central device. This redundancy makes the network more reliable. Local area networks with this redundancy can be modeled using wheels $W_n$, as shown in Figure 11(c). ◀

EXAMPLE 17 **Interconnection Networks for Parallel Computation** For many years, computers executed programs one operation at a time. Consequently, the algorithms written to solve problems were designed to perform one step at a time; such algorithms are called **serial**. (Almost all algorithms described in this book are serial.) However, many computationally intense problems, such as weather simulations, medical imaging, and cryptanalysis, cannot be solved in a reasonable amount of time using serial operations, even on a supercomputer. Furthermore, there is a physical limit to how fast a computer can carry out basic operations, so there will always be problems that cannot be solved in a reasonable length of time using serial operations.

**Parallel processing**, which uses computers made up of many separate processors, each with its own memory, helps overcome the limitations of computers with a single processor. **Parallel algorithms**, which break a problem into a number of subproblems that can be solved

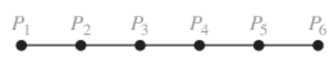



**FIGURE 12 A Linear Array for Six Processors.**

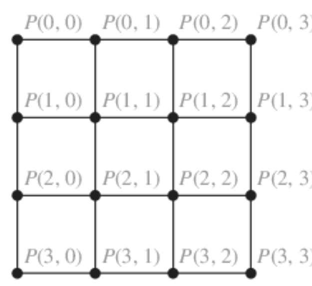



**FIGURE 13 A Mesh Network for 16 Processors.**

concurrently, can then be devised to rapidly solve problems using a computer with multiple processors. In a parallel algorithm, a single instruction stream controls the execution of the algorithm, sending subproblems to different processors, and directs the input and output of these subproblems to the appropriate processors.

When parallel processing is used, one processor may need output generated by another processor. Consequently, these processors need to be interconnected. We can use the appropriate type of graph to represent the interconnection network of the processors in a computer with multiple processors. In the following discussion, we will describe the most commonly used types of interconnection networks for parallel processors. The type of interconnection network used to implement a particular parallel algorithm depends on the requirements for exchange of data between processors, the desired speed, and, of course, the available hardware.

The simplest, but most expensive, network-interconnecting processors include a two-way link between each pair of processors. This network can be represented by $K_n$, the complete graph on $n$ vertices, when there are $n$ processors. However, there are serious problems with this type of interconnection network because the required number of connections is so large. In reality, the number of direct connections to a processor is limited, so when there are a large number of processors, a processor cannot be linked directly to all others. For example, when there are 64 processors, $C(64, 2) = 2016$ connections would be required, and each processor would have to be directly connected to 63 others.

On the other hand, perhaps the simplest way to interconnect $n$ processors is to use an arrangement known as a **linear array**. Each processor $P_i$, other than $P_1$ and $P_n$, is connected to its neighbors $P_{i-1}$ and $P_{i+1}$ via a two-way link. $P_1$ is connected only to $P_2$, and $P_n$ is connected only to $P_{n-1}$. The linear array for six processors is shown in Figure 12. The advantage of a linear array is that each processor has at most two direct connections to other processors. The disadvantage is that it is sometimes necessary to use a large number of intermediate links, called **hops**, for processors to share information.

The **mesh network** (or **two-dimensional array**) is a commonly used interconnection network. In such a network, the number of processors is a perfect square, say $n = m^2$. The $n$ processors are labeled $P(i, j)$, $0 \le i \le m-1$, $0 \le j \le m-1$. Two-way links connect processor $P(i, j)$ with its four neighbors, processors $P(i \pm 1, j)$ and $P(i, j \pm 1)$, as long as these are processors in the mesh. (Note that four processors, on the corners of the mesh, have only two adjacent processors, and other processors on the boundaries have only three neighbors. Sometimes a variant of a mesh network in which every processor has exactly four connections is used; see Exercise 72.) The mesh network limits the number of links for each processor. Communication between some pairs of processors requires $O(\sqrt{n}) = O(m)$ intermediate links. (See Exercise 73.) The graph representing the mesh network for 16 processors is shown in Figure 13.

One important type of interconnection network is the hypercube. For such a network, the number of processors is a power of 2, $n = 2^m$. The $n$ processors are labeled $P_0, P_1, \ldots, P_{n-1}$. Each processor has two-way connections to $m$ other processors. Processor $P_i$ is linked to the processors with indices whose binary representations differ from the binary representation of $i$

in exactly one bit. The hypercube network balances the number of direct connections for each processor and the number of intermediate connections required so that processors can communicate. Many computers have been built using a hypercube network, and many parallel algorithms have been devised that use a hypercube network. The graph $Q_m$, the $m$-cube, represents the hypercube network with $n = 2^m$ processors. Figure 14 displays the hypercube network for eight processors. (Figure 14 displays a different way to draw $Q_3$ than was shown in Figure 6.) ◀

## 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 New York, Denver, Detroit, and Atlanta. Then we can ignore the other computer centers and all telephone lines not linking two of these specific four computer centers. In the graph model for the large network, we can remove the vertices corresponding to the computer centers other than the four of interest, and we can remove all edges incident with a vertex that was removed. When edges and vertices are removed from a graph, without removing endpoints of any remaining edges, a smaller graph is obtained. Such a graph is called a **subgraph** of the original graph.

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 $H \neq G$.

Given a set of vertices of a graph, we can form a subgraph of this graph with these vertices and the edges of the graph that connect them.

DEFINITION 8 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 of this edge are in $W$.

EXAMPLE 18 The graph $G$ shown in Figure 15 is a subgraph of $K_5$. If we add the edge connecting $c$ and $e$ to $G$, we obtain the subgraph induced by $W = \{a, b, c, e\}$. ◀

REMOVING OR ADDING EDGES OF A GRAPH Given a graph $G = (V, E)$ and an edge $e \in E$, we can produce a subgraph of $G$ by removing the edge $e$. The resulting subgraph, denoted by $G - e$, has the same vertex set $V$ as $G$. Its edge set is $E - e$. Hence,

$$G - e = (V, E - \{e\}).$$

Similarly, if $E'$ is a subset of $E$, we can produce a subgraph of $G$ by removing the edges in $E'$ from the graph. The resulting subgraph has the same vertex set $V$ as $G$. Its edge set is $E - E'$.

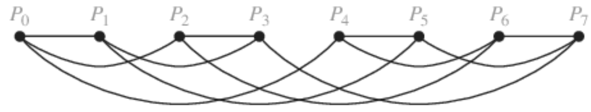



FIGURE 14 **A Hypercube Network for Eight Processors.**

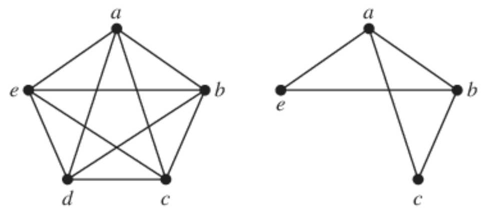



FIGURE 15 **A Subgraph of $K_5$.**

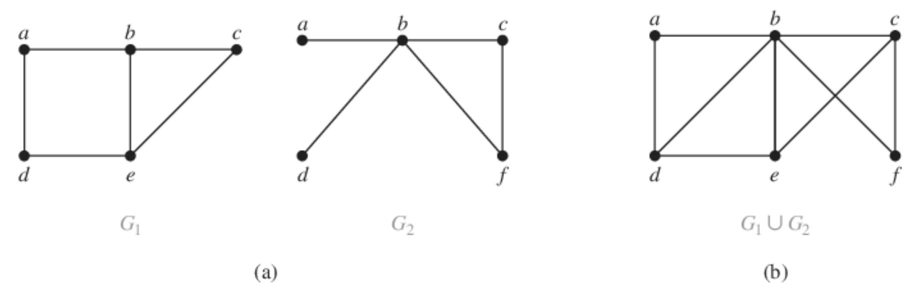



FIGURE 16 **(a) The Simple Graphs $G_1$ and $G_2$; (b) Their Union $G_1 \cup G_2$.**

We can also add an edge $e$ to a graph to produce a new larger graph when this edge connects two vertices already in $G$. We denote by $G + e$ the new graph produced by adding a new edge $e$, connecting two previously nonincident vertices, to the graph $G$ Hence,

$$G + e = (V, E \cup \{e\}).$$

The vertex set of $G + e$ is the same as the vertex set of $G$ and the edge set is the union of the edge set of $G$ and the set $\{e\}$.

EDGE CONTRACTIONS Sometimes when we remove an edge from a graph, we do not want to retain the endpoints of this edge as separate vertices in the resulting subgraph. In such a case we perform an **edge contraction** which removes an edge $e$ with endpoints $u$ and $v$ and merges $u$ and $w$ into a new single vertex $w$, and for each edge with $u$ or $v$ as an endpoint replaces the edge with one with $w$ as endpoint in place of $u$ or $v$ and with the same second endpoint. Hence, the contraction of the edge $e$ with endpoints $u$ and $v$ in the graph $G = (V, E)$ produces a new graph $G' = (V', E')$ (which is not a subgraph of $G$), where $V' = V - \{u, v\} \cup \{w\}$ and $E'$ contains the edges in $E$ which do not have either $u$ or $v$ as endpoints and an edge connecting $w$ to every neighbor of either $u$ or $v$ in $V$. For example, the contraction of the edge connecting the vertices $e$ and $c$ in the graph $G_1$ in Figure 16 produces a new graph $G_1'$ with vertices $a$, $b$, $d$, and $w$. As in $G_1$, there is an edge in $G_1'$ connecting $a$ and $b$ and an edge connecting $a$ and $d$. There also is an edge in $G_1'$ that connects $b$ and $w$ that replaces the edges connecting $b$ and $c$ and connecting $b$ and $e$ in $G_1$ and an edge in $G_1'$ that connects $d$ and $w$ replacing the edge connecting $d$ and $e$ in $G_1$.

REMOVING VERTICES FROM A GRAPH When we remove a vertex $v$ and all edges incident to it from $G = (V, E)$, we produce a subgraph, denoted by $G - v$. Observe that $G - v = (V - v, E')$, where $E'$ is the set of edges of $G$ not incident to $v$. Similarly, if $V'$ is a subset of $V$, then the graph $G - V'$ is the subgraph $(V - V', E')$, where $E'$ is the set of edges of $G$ not incident to a vertex in $V'$.

GRAPH UNIONS Two or more graphs can be combined in various ways. The new graph that contains all the vertices and edges of these graphs is called the **union** of the graphs. We will give a more formal definition for the union of two simple graphs.

DEFINITION 9 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 $G_1 \cup G_2$.

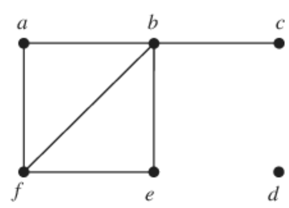
EXAMPLE 19 Find the union of the graphs $G_1$ and $G_2$ shown in Figure 16(a). ◀

*Solution:* The vertex set of the union $G_1 \cup G_2$ is the union of the two vertex sets, namely, $\{a, b, c, d, e, f\}$. The edge set of the union is the union of the two edge sets. The union is displayed in Figure 16(b).

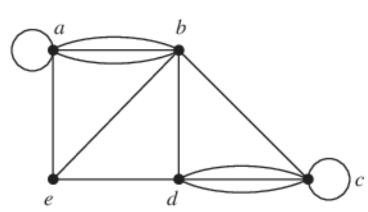
## Exercises

In Exercises 1–3 find the number of vertices, the number of edges, and the degree of each vertex in the given undirected graph. Identify all isolated and pendant vertices.

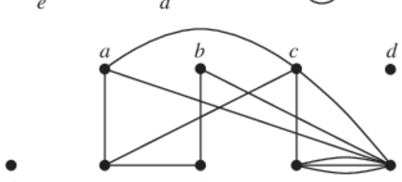
**1.**




**2.**




**3.**



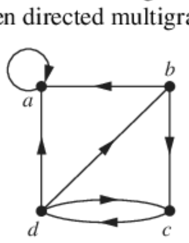

**4.** Find the sum of the degrees of the vertices of each graph in Exercises 1–3 and verify that it equals twice the number of edges in the graph.

**5.** Can a simple graph exist with 15 vertices each of degree five?

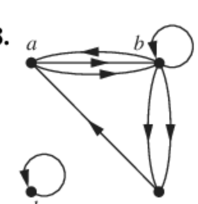
**6.** Show that the sum, over the set of people at a party, of the number of people a person has shaken hands with, is even. Assume that no one shakes his or her own hand.

In Exercises 7–9 determine the number of vertices and edges and find the in-degree and out-degree of each vertex for the given directed multigraph.

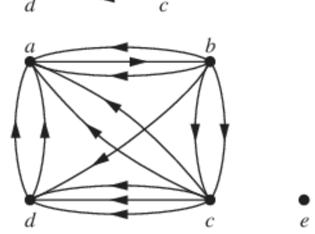
**7.**




**8.**




**9.**




**10.** For each of the graphs in Exercises 7–9 determine the sum of the in-degrees of the vertices and the sum of the out-degrees of the vertices directly. Show that they are both equal to the number of edges in the graph.

**11.** Construct the underlying undirected graph for the graph with directed edges in Figure 2.

**12.** What does the degree of a vertex represent in the acquaintanceship graph, where vertices represent all the people in the world? What does the neighborhood a vertex in this graph represent? What do isolated and pendant vertices in this graph represent? In one study it was estimated that the average degree of a vertex in this graph is 1000. What does this mean in terms of the model?

**13.** What does the degree of a vertex represent in an academic collaboration graph? What does the neighborhood of a vertex represent? What do isolated and pendant vertices represent?

**14.** What does the degree of a vertex in the Hollywood graph represent? What does the neighborhood of a vertex represent? What do the isolated and pendant vertices represent?

**15.** What do the in-degree and the out-degree of a vertex in a telephone call graph, as described in Example 4 of Section 10.1, represent? What does the degree of a vertex in the undirected version of this graph represent?

**16.** What do the in-degree and the out-degree of a vertex in the Web graph, as described in Example 5 of Section 10.1, represent?

**17.** What do the in-degree and the out-degree of a vertex in a directed graph modeling a round-robin tournament represent?

**18.** Show that in a simple graph with at least two vertices there must be two vertices that have the same degree.

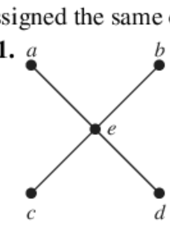
**19.** Use Exercise 18 to show that in a group of people, there must be two people who are friends with the same number of other people in the group.

**20.** Draw these graphs.

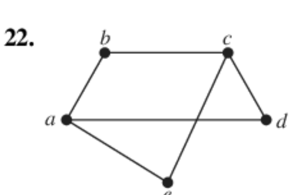
**a)** $K_7$ **b)** $K_{1,8}$ **c)** $K_{4,4}$
**d)** $C_7$ **e)** $W_7$ **f)** $Q_4$

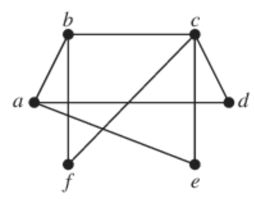
In Exercises 21–25 determine whether the graph is bipartite. You may find it useful to apply Theorem 4 and answer the question by determining whether it is possible to assign either red or blue to each vertex so that no two adjacent vertices are assigned the same color.

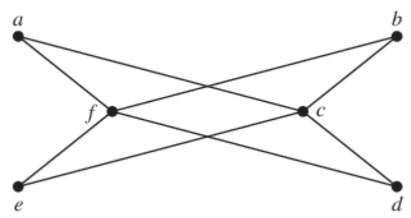
**21.**

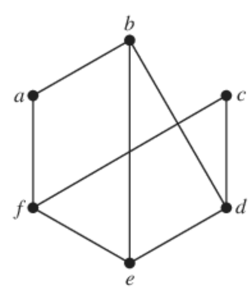



**22.**



**23.** 


**24.** 


**25.** 


**26.** For which values of $n$ are these graphs bipartite?

**a)** $K_n$ **b)** $C_n$ **c)** $W_n$ **d)** $Q_n$

**27.** Suppose that there are four employees in the computer support group of the School of Engineering of a large university. Each employee will be assigned to support one of four different areas: hardware, software, networking, and wireless. Suppose that Ping is qualified to support hardware, networking, and wireless; Quiggley is qualified to support software and networking; Ruiz is qualified to support networking and wireless, and Sitea is qualified to support hardware and software.

**a)** Use a bipartite graph to model the four employees and their qualifications.

**b)** Use Hall's theorem to determine whether there is an assignment of employees to support areas so that each employee is assigned one area to support.

**c)** If an assignment of employees to support areas so that each employee is assigned to one support area exists, find one.

**28.** Suppose that a new company has five employees: Zamora, Agraharam, Smith, Chou, and Macintyre. Each employee will assume one of six responsiblities: planning, publicity, sales, marketing, development, and industry relations. Each employee is capable of doing one or more of these jobs: Zamora could do planning, sales, marketing, or industry relations; Agraharam could do planning or development; Smith could do publicity, sales, or industry relations; Chou could do planning, sales, or industry relations; and Macintyre could do planning, publicity, sales, or industry relations.

**a)** Model the capabilities of these employees using a bipartite graph.

**b)** Find an assignment of responsibilites such that each employee is assigned one responsibility.

**c)** Is the matching of responsibilities you found in part (b) a complete matching? Is it a maximum matching?

**29.** Suppose that there are five young women and five young men on an island. Each man is willing to marry some of the women on the island and each woman is willing to marry any man who is willing to marry her. Suppose that Sandeep is willing to marry Tina and Vandana; Barry is willing to marry Tina, Xia, and Uma; Teja is willing to marry Tina and Zelda; Anil is willing to marry Vandana and Zelda; and Emilio is willing to marry Tina and Zelda. Use Hall's theorem to show there is no matching of the young men and young women on the island such that each young man is matched with a young woman he is willing to marry.

**30.** Suppose that there are five young women and six young men on an island. Each woman is willing to marry some of the men on the island and each man is willing to marry any woman who is willing to marry him. Suppose that Anna is willing to marry Jason, Larry, and Matt; Barbara is willing to marry Kevin and Larry; Carol is willing to marry Jason, Nick, and Oscar; Diane is willing to marry Jason, Larry, Nick, and Oscar; and Elizabeth is willing to marry Jason and Matt.

**a)** Model the possible marriages on the island using a bipartite graph.

**b)** Find a matching of the young women and the young men on the island such that each young woman is matched with a young man whom she is willing to marry.

**c)** Is the matching you found in part (b) a complete matching? Is it a maximum matching?

***31.** Suppose there is an integer $k$ such that every man on a desert island is willing to marry exactly $k$ of the women on the island and every woman on the island is willing to marry exactly $k$ of the men. Also, suppose that a man is willing to marry a woman if and only if she is willing to marry him. Show that it is possible to match the men and women on the island so that everyone is matched with someone that they are willing to marry.

***32.** In this exercise we prove a theorem of Øystein Ore. Suppose that $G = (V, E)$ is a bipartite graph with bipartition $(V_1, V_2)$ and that $A \subseteq V_1$. Show that the maximum number of vertices of $V_1$ that are the endpoints of a matching of $G$ equals $|V_1| - \max_{A \subseteq V_1} \operatorname{def}(A)$, where $\operatorname{def}(A) = |A| - |N(A)|$. (Here, $\operatorname{def}(A)$ is called the **deficiency** of $A$.) [*Hint:* Form a larger graph by adding $\max_{A \subseteq V_1} \operatorname{def}(A)$ new vertices to $V_2$ and connect all of them to the vertices of $V_1$.]

**33.** For the graph $G$ in Exercise 1 find

**a)** the subgraph induced by the vertices $a$, $b$, $c$, and $f$.

**b)** the new graph $G_1$ obtained from $G$ by contracting the edge connecting $b$ and $f$.

**34.** Let $n$ be a positive integer. Show that a subgraph induced by a nonempty subset of the vertex set of $K_n$ is a complete graph.

**35.** How many vertices and how many edges do these graphs have?

**a)** $K_n$ **b)** $C_n$ **c)** $W_n$
**d)** $K_{m,n}$ **e)** $Q_n$

The **degree sequence** of a graph is the sequence of the degrees of the vertices of the graph in nonincreasing order. For example, the degree sequence of the graph $G$ in Example 1 is 4, 4, 4, 3, 2, 1, 0.

**36.** Find the degree sequences for each of the graphs in Exercises 21–25.

**37.** Find the degree sequence of each of the following graphs.

**a)** $K_4$ **b)** $C_4$ **c)** $W_4$
**d)** $K_{2,3}$ **e)** $Q_3$

**38.** What is the degree sequence of the bipartite graph $K_{m,n}$ where $m$ and $n$ are positive integers? Explain your answer.

**39.** What is the degree sequence of $K_n$, where $n$ is a positive integer? Explain your answer.

**40.** How many edges does a graph have if its degree sequence is 4, 3, 3, 2, 2? Draw such a graph.

**41.** How many edges does a graph have if its degree sequence is 5, 2, 2, 2, 2, 1? Draw such a graph.

A sequence $d_1, d_2, \ldots, d_n$ is called **graphic** if it is the degree sequence of a simple graph.

**42.** Determine whether each of these sequences is graphic. For those that are, draw a graph having the given degree sequence.

**a)** 5, 4, 3, 2, 1, 0 **b)** 6, 5, 4, 3, 2, 1 **c)** 2, 2, 2, 2, 2, 2
**d)** 3, 3, 3, 2, 2, 2 **e)** 3, 3, 2, 2, 2, 2 **f)** 1, 1, 1, 1, 1, 1
**g)** 5, 3, 3, 3, 3, 3 **h)** 5, 5, 4, 3, 2, 1

**43.** Determine whether each of these sequences is graphic. For those that are, draw a graph having the given degree sequence.

**a)** 3, 3, 3, 3, 2 **b)** 5, 4, 3, 2, 1 **c)** 4, 4, 3, 2, 1
**d)** 4, 4, 3, 3, 3 **e)** 3, 2, 2, 1, 0 **f)** 1, 1, 1, 1, 1

***44.** Suppose that $d_1, d_2, \ldots, d_n$ is a graphic sequence. Show that there is a simple graph with vertices $v_1, v_2, \ldots, v_n$ such that $\deg(v_i) = d_i$ for $i = 1, 2, \ldots, n$ and $v_1$ is adjacent to $v_2, \ldots, v_{d_1+1}$.

***45.** Show that a sequence $d_1, d_2, \ldots, d_n$ of nonnegative integers in nonincreasing order is a graphic sequence if and only if the sequence obtained by reordering the terms of the sequence $d_2 - 1, \ldots, d_{d_1+1} - 1, d_{d_1+2}, \ldots, d_n$ so that the terms are in nonincreasing order is a graphic sequence.

***46.** Use Exercise 45 to construct a recursive algorithm for determining whether a nonincreasing sequence of positive integers is graphic.

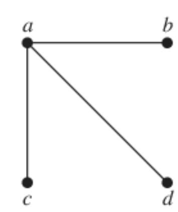
**47.** Show that every nonincreasing sequence of nonnegative integers with an even sum of its terms is the degree sequence of a pseudograph, that is, an undirected graph where loops are allowed. [*Hint:* Construct such a graph by first adding as many loops as possible at each vertex. Then add additional edges connecting vertices of odd degree. Explain why this construction works.]

**48.** How many subgraphs with at least one vertex does $K_2$ have?

**49.** How many subgraphs with at least one vertex does $K_3$ have?

**50.** How many subgraphs with at least one vertex does $W_3$ have?

**51.** Draw all subgraphs of this graph.




**52.** Let $G$ be a graph with $v$ vertices and $e$ edges. Let $M$ be the maximum degree of the vertices of $G$, and let $m$ be the minimum degree of the vertices of $G$. Show that

**a)** $2e/v \geq m$. **b)** $2e/v \leq M$.

A simple graph is called **regular** if every vertex of this graph has the same degree. A regular graph is called $n$-**regular** if every vertex in this graph has degree $n$.

**53.** For which values of $n$ are these graphs regular?

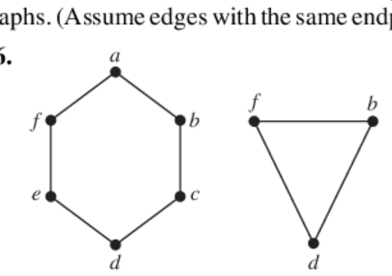
**a)** $K_n$ **b)** $C_n$ **c)** $W_n$ **d)** $Q_n$

**54.** For which values of $m$ and $n$ is $K_{m,n}$ regular?

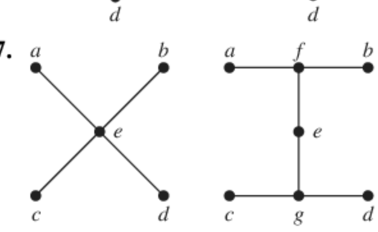
**55.** How many vertices does a regular graph of degree four with 10 edges have?

In Exercises 56–58 find the union of the given pair of simple graphs. (Assume edges with the same endpoints are the same.)

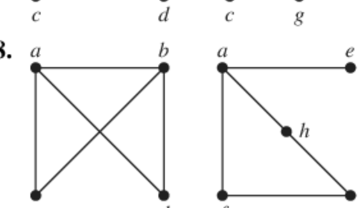
**56.**




**57.**




**58.**




**59.** The **complementary graph** $\overline{G}$ of a simple graph $G$ has the same vertices as $G$. Two vertices are adjacent in $\overline{G}$ if and only if they are not adjacent in $G$. Describe each of these graphs.

**a)** $\overline{K_n}$ **b)** $\overline{K_{m,n}}$ **c)** $\overline{C_n}$ **d)** $\overline{Q_n}$

**60.** If $G$ is a simple graph with 15 edges and $\overline{G}$ has 13 edges, how many vertices does $G$ have?

61. If the simple graph $G$ has $v$ vertices and $e$ edges, how many edges does $\overline{G}$ have?

62. If the degree sequence of the simple graph $G$ is $4, 3, 3, 2, 2$, what is the degree sequence of $\overline{G}$?

63. If the degree sequence of the simple graph $G$ is $d_1, d_2, \ldots, d_n$, what is the degree sequence of $\overline{G}$?

*64. Show that if $G$ is a bipartite simple graph with $v$ vertices and $e$ edges, then $e \leq v^2/4$.

65. Show that if $G$ is a simple graph with $n$ vertices, then the union of $G$ and $\overline{G}$ is $K_n$.

*66. Describe an algorithm to decide whether a graph is bipartite based on the fact that a graph is bipartite if and only if it is possible to color its vertices two different colors so that no two vertices of the same color are adjacent.

The **converse** of a directed graph $G = (V, E)$, denoted by $G^{conv}$, is the directed graph $(V, F)$, where the set $F$ of edges of $G^{conv}$ is obtained by reversing the direction of each edge in $E$.

67. Draw the converse of each of the graphs in Exercises 7–9 in Section 10.1.

68. Show that $(G^{conv})^{conv} = G$ whenever $G$ is a directed graph.

69. Show that the graph $G$ is its own converse if and only if the relation associated with $G$ (see Section 9.3) is symmetric.

70. Show that if a bipartite graph $G = (V, E)$ is $n$-regular for some positive integer $n$ (see the preamble to Exercise 53) and $(V_1, V_2)$ is a bipartition of $V$, then $|V_1| = |V_2|$. That is, show that the two sets in a bipartition of the vertex set of an $n$-regular graph must contain the same number of vertices.

71. Draw the mesh network for interconnecting nine parallel processors.

72. In a variant of a mesh network for interconnecting $n = m^2$ processors, processor $P(i, j)$ is connected to the four processors $P((i \pm 1) \textbf{ mod } m, j)$ and $P(i, (j \pm 1) \textbf{ mod } m)$, so that connections wrap around the edges of the mesh. Draw this variant of the mesh network for 16 processors.

73. Show that every pair of processors in a mesh network of $n = m^2$ processors can communicate using $O(\sqrt{n}) = O(m)$ hops between directly connected processors.

# 10.3 Representing Graphs and Graph Isomorphism

## Introduction

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

Sometimes, two graphs have exactly the same form, in the sense that there is a one-to-one correspondence between their vertex sets that preserves edges. In such a case, we say that the two graphs are **isomorphic**. Determining whether two graphs are isomorphic is an important problem of graph theory that we will study in this section.

## Representing Graphs

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.

EXAMPLE 1 Use adjacency lists to describe the simple graph given in Figure 1.

*Solution:* Table 1 lists those vertices adjacent to each of the vertices of the graph. ◀

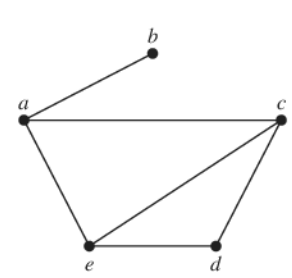



FIGURE 1 **A Simple Graph.**

TABLE 1 **An Adjacency List for a Simple Graph.**

| *Vertex* | *Adjacent Vertices* |
|---|---|
| $a$ | $b, c, e$ |
| $b$ | $a$ |
| $c$ | $a, d, e$ |
| $d$ | $c, e$ |
| $e$ | $a, c, d$ |

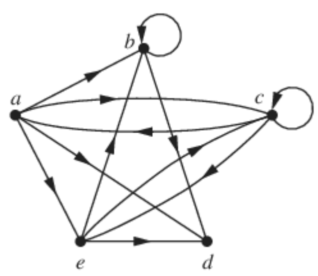



FIGURE 2 **A Directed Graph.**

**TABLE 2 An Adjacency List for a Directed Graph.**

| *Initial Vertex* | *Terminal Vertices* |
|---|---|
| $a$ | $b, c, d, e$ |
| $b$ | $b, d$ |
| $c$ | $a, c, e$ |
| $d$ | |
| $e$ | $b, c, d$ |

**EXAMPLE 2** Represent the directed graph shown in Figure 2 by listing all the vertices that are the terminal vertices of edges starting at each vertex of the graph.

*Solution:* Table 2 represents the directed graph shown in Figure 2. ◄

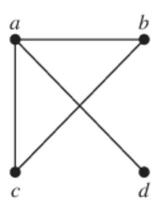



FIGURE 3
**Simple Graph.**

## Adjacency Matrices

Carrying out graph algorithms using the representation of graphs by lists of edges, or by adjacency lists, can be cumbersome if there are many edges in the graph. To simplify computation, graphs can be represented using matrices. Two types of matrices commonly used to represent graphs will be presented here. One is based on the adjacency of vertices, and the other is based on incidence of vertices and edges.

Links

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 $n$ x $n$ 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

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

**EXAMPLE 3** Use an adjacency matrix to represent the graph shown in Figure 3.

*Solution:* We order the vertices as $a, b, c, d$. The matrix representing this graph is

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

◄

**EXAMPLE 4** Draw a graph with the adjacency matrix

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

with respect to the ordering of vertices $a, b, c, d$.

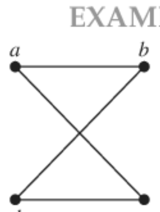



FIGURE 4
**A Graph with the Given Adjacency Matrix.**

*Solution:* A graph with this adjacency matrix is shown in Figure 4. ◄

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.

The adjacency matrix of a simple graph is symmetric, that is, $a_{ij} = a_{ji}$, because both of these entries are 1 when $v_i$ and $v_j$ are adjacent, and both are 0 otherwise. Furthermore, because a simple graph has no loops, each entry $a_{ii}$, $i = 1, 2, 3, \ldots, n$, is 0.

Adjacency matrices can also be used to represent undirected graphs with loops and with multiple edges. A loop at the vertex $v_i$ is represented by a 1 at the $(i, i)$th position of the adjacency matrix. When multiple edges connecting the same pair of vertices $v_i$ and $v_j$, or multiple loops at the same vertex, are present, the adjacency matrix is no longer a zero–one matrix, because the $(i, j)$th entry of this matrix equals the number of edges that are associated to $\{v_i, v_j\}$. All undirected graphs, including multigraphs and pseudographs, have symmetric adjacency matrices.

**EXAMPLE 5** Use an adjacency matrix to represent the pseudograph shown in Figure 5.

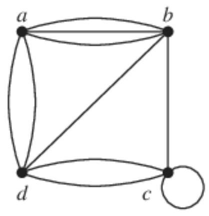



**FIGURE 5**
**A Pseudograph.**

*Solution:* The adjacency matrix using the ordering of vertices $a, b, c, d$ is

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

◀

We used zero–one matrices in Chapter 9 to represent directed graphs. The matrix for a directed graph $G = (V, E)$ has a 1 in its $(i, j)$th position if there is an edge from $v_i$ to $v_j$, where $v_1, v_2, \ldots, v_n$ is an arbitrary listing of the vertices of the directed graph. In other words, if $\mathbf{A} = [a_{ij}]$ is the adjacency matrix for the directed graph with respect to this listing of the vertices, then

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

The adjacency matrix for a directed graph does not have to be symmetric, because there may not be an edge from $v_j$ to $v_i$ when there is an edge from $v_i$ to $v_j$.

Adjacency matrices can also be used to represent directed multigraphs. Again, such matrices are not zero–one matrices when there are multiple edges in the same direction connecting two vertices. In the adjacency matrix for a directed multigraph, $a_{ij}$ equals the number of edges that are associated to $(v_i, v_j)$.

TRADE-OFFS BETWEEN ADJACENCY LISTS AND ADJACENCY MATRICES When a simple graph contains relatively few edges, that is, when it is **sparse**, it is usually preferable to use adjacency lists rather than an adjacency matrix to represent the graph. For example, if each vertex has degree not exceeding $c$, where $c$ is a constant much smaller than $n$, then each adjacency list contains $c$ or fewer vertices. Hence, there are no more than $cn$ items in all these adjacency lists. On the other hand, the adjacency matrix for the graph has $n^2$ entries. Note, however, that the adjacency matrix of a sparse graph is a **sparse matrix**, that is, a matrix with few nonzero entries, and there are special techniques for representing, and computing with, sparse matrices.

Now suppose that a simple graph is **dense**, that is, suppose that it contains many edges, such as a graph that contains more than half of all possible edges. In this case, using an adjacency matrix to represent the graph is usually preferable over using adjacency lists. To see why, we compare the complexity of determining whether the possible edge $\{v_i, v_j\}$ is present. Using an adjacency matrix, we can determine whether this edge is present by examining the $(i, j)$th entry in the matrix. This entry is 1 if the graph contains this edge and is 0 otherwise. Consequently, we need make only one comparison, namely, comparing this entry with 0, to determine whether this edge is present. On the other hand, when we use adjacency lists to represent the graph, we need to search the list of vertices adjacent to either $v_i$ or $v_j$ to determine whether this edge is present. This can require $\Theta(|V|)$ comparisons when many edges are present.

## Incidence Matrices

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

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

EXAMPLE 6 Represent the graph shown in Figure 6 with an incidence matrix.

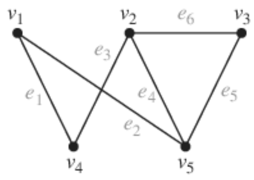



FIGURE 6 **An Undirected Graph.**

*Solution:* The incidence matrix is

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

◀

Incidence matrices can also be used to represent multiple edges and loops. Multiple edges are represented in the incidence matrix using columns with identical entries, because these edges are incident with the same pair of vertices. Loops are represented using a column with exactly one entry equal to 1, corresponding to the vertex that is incident with this loop.

EXAMPLE 7 Represent the pseudograph shown in Figure 7 using an incidence matrix.

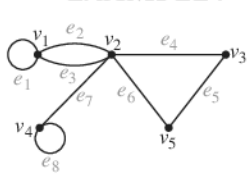



FIGURE 7 **A Pseudograph.**

*Solution:* The incidence matrix for this graph is

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

◀

## Isomorphism of Graphs

We often need to know whether it is possible to draw two graphs in the same way. That is, do the graphs have the same structure when we ignore the identities of their vertices? For instance, in chemistry, graphs are used to model chemical compounds (in a way we will describe later). Different compounds can have the same molecular formula but can differ in structure. Such compounds can be represented by graphs that cannot be drawn in the same way. The graphs representing previously known compounds can be used to determine whether a supposedly new compound has been studied before.

There is a useful terminology for graphs with the same structure.

**DEFINITION 1** 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*.

In other words, 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 simple graphs is an equivalence relation. (We leave the verification of this as Exercise 45.)

**EXAMPLE 8** Show that the graphs $G = (V, E)$ and $H = (W, F)$, displayed in Figure 8, are isomorphic.

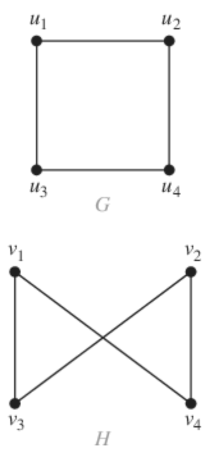



**FIGURE 8 The Graphs $G$ and $H$.**

*Solution:* 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 $W$. To see that this correspondence preserves adjacency, note that adjacent vertices in $G$ are $u_1$ and $u_2$, $u_1$ and $u_3$, $u_2$ and $u_4$, and $u_3$ and $u_4$, and each of the pairs $f(u_1) = v_1$ and $f(u_2) = v_4$, $f(u_1) = v_1$ and $f(u_3) = v_3$, $f(u_2) = v_4$ and $f(u_4) = v_2$, and $f(u_3) = v_3$ and $f(u_4) = v_2$ consists of two adjacent vertices in $H$. ◀

## Determining whether Two Simple Graphs are Isomorphic

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.

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, but that is preserved by isomorphism. A property preserved by isomorphism of graphs is called a **graph invariant**. For instance, isomorphic simple graphs must have the same number of vertices, because there is a one-to-one correspondence between the sets of vertices of the graphs.

Isomorphic simple graphs also must have the same number of edges, because the one-to-one correspondence between vertices establishes a one-to-one correspondence between edges. In addition, the degrees of the vertices in isomorphic simple graphs must be the same. That is, a vertex $v$ of degree $d$ in $G$ must correspond to a vertex $f(v)$ of degree $d$ in $H$, because a vertex $w$ in $G$ is adjacent to $v$ if and only if $f(v)$ and $f(w)$ are adjacent in $H$.

**EXAMPLE 9** Show that the graphs displayed in Figure 9 are not isomorphic.

*Solution:* Both $G$ and $H$ have five vertices and six edges. However, $H$ 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. ◀

The number of vertices, the number of edges, and the number of vertices of each degree are all invariants under isomorphism. If any of these quantities differ in two simple graphs, these graphs cannot be isomorphic. However, when these invariants are the same, it does not necessarily mean that the two graphs are isomorphic. There are no useful sets of invariants currently known that can be used to determine whether simple graphs are isomorphic.

*The word *isomorphism* comes from the Greek roots *isos* for "equal" and *morphe* for "form."

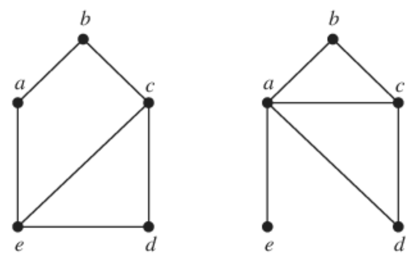



FIGURE 9 **The Graphs $G$ and $H$.**

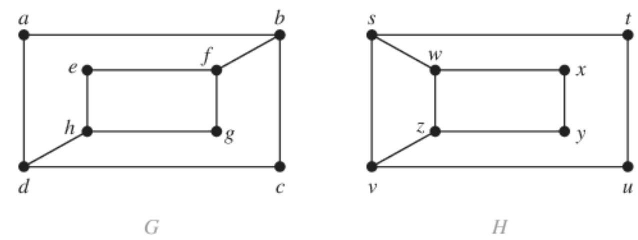



FIGURE 10 **The Graphs $G$ and $H$.**

EXAMPLE 10 Determine whether the graphs shown in Figure 10 are isomorphic.

*Solution:* The graphs $G$ and $H$ both have eight vertices and 10 edges. They also both have four vertices of degree two and four of degree three. Because these invariants all agree, it is still conceivable that these graphs are isomorphic.

However, $G$ and $H$ are not isomorphic. To see this, note that because $\deg(a) = 2$ in $G$, $a$ must correspond to either $t$, $u$, $x$, or $y$ 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 $G$.

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 (the reader should verify this). However, these subgraphs, shown in Figure 11, are not isomorphic. ◀

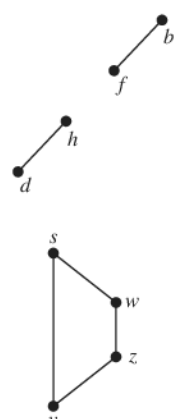



FIGURE 11 **The Subgraphs of $G$ and $H$ Made Up of Vertices of Degree Three and the Edges Connecting Them.**

To show that a function $f$ from the vertex set of a graph $G$ to the vertex set of a graph $H$ is an isomorphism, we need to show that $f$ preserves the presence and absence of edges. One helpful way to do this is to use adjacency matrices. In particular, to show that $f$ is an isomorphism, we can show that the adjacency matrix of $G$ is the same as the adjacency matrix of $H$, when rows and columns are labeled to correspond to the images under $f$ of the vertices in $G$ that are the labels of these rows and columns in the adjacency matrix of $G$. We illustrate how this is done in Example 11.

EXAMPLE 11 Determine whether the graphs $G$ and $H$ displayed in Figure 12 are isomorphic.

*Solution:* Both $G$ and $H$ have six vertices and seven edges. Both have four vertices of degree two and two vertices of degree three. It is also easy to see that the subgraphs of $G$ and $H$ consisting of all vertices of degree two and the edges connecting them are isomorphic (as the reader should verify). Because $G$ and $H$ agree with respect to these invariants, it is reasonable to try to find an isomorphism $f$.

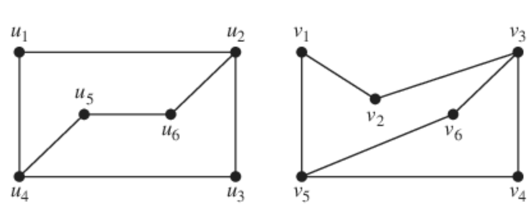



FIGURE 12 **Graphs $G$ and $H$.**

We now will define a function $f$ and then determine whether it is an isomorphism. Because $\deg(u_1) = 2$ and because $u_1$ is not adjacent to any other vertex of degree two, the image of $u_1$ must be either $v_4$ or $v_6$, the only vertices of degree two in $H$ not adjacent to a vertex of degree two. We arbitrarily set $f(u_1) = v_6$. [If we found that this choice did not lead to isomorphism, we would then try $f(u_1) = v_4$.] Because $u_2$ is adjacent to $u_1$, the possible images of $u_2$ are $v_3$ and $v_5$. We arbitrarily set $f(u_2) = v_3$. Continuing in this way, using adjacency of vertices and degrees as a guide, we set $f(u_3) = v_4$, $f(u_4) = v_5$, $f(u_5) = v_1$, and $f(u_6) = v_2$. We now have a one-to-one correspondence between the vertex set of $G$ and the vertex set of $H$, namely, $f(u_1) = v_6$, $f(u_2) = v_3$, $f(u_3) = v_4$, $f(u_4) = v_5$, $f(u_5) = v_1$, $f(u_6) = v_2$. To see whether $f$ preserves edges, we examine the adjacency matrix of $G$,

$$\mathbf{A}_G = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{array}\!\!\begin{array}{c} \begin{array}{cccccc} u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array},$$

and the adjacency matrix of $H$ with the rows and columns labeled by the images of the corresponding vertices in $G$,

$$\mathbf{A}_H = \begin{array}{c} \\ v_6 \\ v_3 \\ v_4 \\ v_5 \\ v_1 \\ v_2 \end{array}\!\!\begin{array}{c} \begin{array}{cccccc} v_6 & v_3 & v_4 & v_5 & v_1 & v_2 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array}.$$

Because $\mathbf{A}_G = \mathbf{A}_H$, it follows that $f$ preserves edges. We conclude that $f$ is an isomorphism, so $G$ and $H$ are isomorphic. Note that if $f$ turned out not to be an isomorphism, we would *not* have established that $G$ and $H$ are not isomorphic, because another correspondence of the vertices in $G$ and $H$ may be an isomorphism. ◀

ALGORITHMS FOR GRAPH ISOMORPHISM The best algorithms known for determining whether two graphs are isomorphic have exponential worst-case time complexity (in the number of vertices of the graphs). However, linear average-case time complexity algorithms are known that solve this problem, and there is some hope, but also skepticism, that an algorithm with polynomial worst-case time complexity for determining whether two graphs are isomorphic can be found. The best practical general purpose software for isomorphism testing, called NAUTY, can be used to determine whether two graphs with as many as 100 vertices are isomorphic in less than a second on a modern PC. NAUTY software can be downloaded over the Internet and experimented with. Practical algorithms for determining whether two graphs are isomorphic exist for graphs that are restricted in various ways, such as when the maximum degree of vertices is small. The problem of determining whether any two graphs are isomorphic is of special interest because it is one of only a few NP problems (see Exercise 72) not known to be either tractable or NP-complete (see Section 3.3).

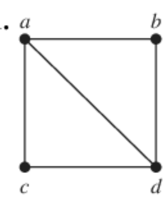
APPLICATIONS OF GRAPH ISOMORPHISMS Graph isomorphisms, and functions that are almost graph isomorphisms, arise in applications of graph theory to chemistry and to the design of electronic circuits, and other areas including bioinformatics and computer vision.

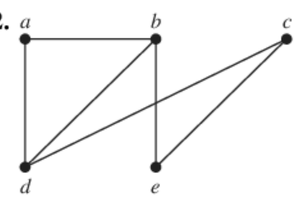
Chemists use multigraphs, known as molecular graphs, to model chemical compounds. In these graphs, vertices represent atoms and edges represent chemical bonds between these atoms. Two structural isomers, molecules with identical molecular formulas but with atoms bonded differently, have nonisomorphic molecular graphs. When a potentially new chemical compound is synthesized, a database of molecular graphs is checked to see whether the molecular graph of the compound is the same as one already known.

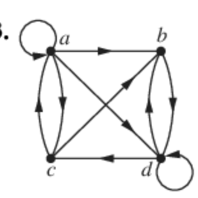
Electronic circuits are modeled using graphs in which vertices represent components and edges represent connections between them. Modern integrated circuits, known as chips, are miniaturized electronic circuits, often with millions of transistors and connections between them. Because of the complexity of modern chips, automation tools are used to design them. Graph isomorphism is the basis for the verification that a particular layout of a circuit produced by an automated tool corresponds to the original schematic of the design. Graph isomorphism can also be used to determine whether a chip from one vendor includes intellectual property from a different vendor. This can be done by looking for large isomorphic subgraphs in the graphs modeling these chips.

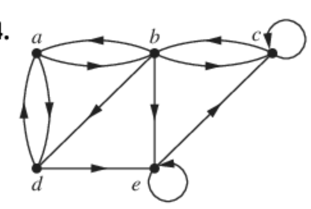
## Exercises

In Exercises 1–4 use an adjacency list to represent the given graph.

**1.**



**2.**



**3.**



**4.**



**5.** Represent the graph in Exercise 1 with an adjacency matrix.

**6.** Represent the graph in Exercise 2 with an adjacency matrix.

**7.** Represent the graph in Exercise 3 with an adjacency matrix.

**8.** Represent the graph in Exercise 4 with an adjacency matrix.

**9.** Represent each of these graphs with an adjacency matrix.

**a)** $K_4$ **b)** $K_{1,4}$ **c)** $K_{2,3}$
**d)** $C_4$ **e)** $W_4$ **f)** $Q_3$

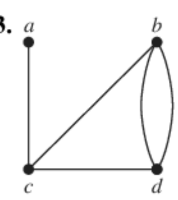
In Exercises 10–12 draw a graph with the given adjacency matrix.

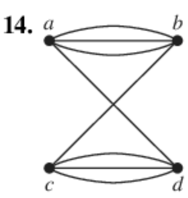
**10.** $\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

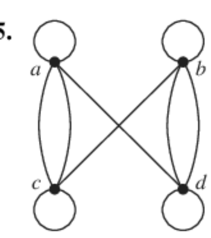
**11.** $\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$

**12.** $\begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \end{bmatrix}$

In Exercises 13–15 represent the given graph using an adjacency matrix.

**13.**



**14.**



**15.**



In Exercises 16–18 draw an undirected graph represented by the given adjacency matrix.

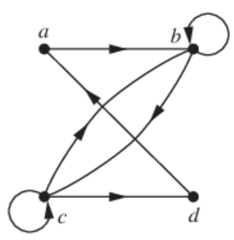
**16.** $\begin{bmatrix} 1 & 3 & 2 \\ 3 & 0 & 4 \\ 2 & 4 & 0 \end{bmatrix}$

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

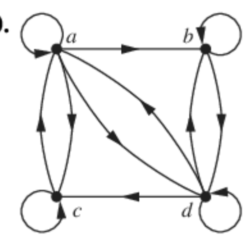
**18.** $\begin{bmatrix} 0 & 1 & 3 & 0 & 4 \\ 1 & 2 & 1 & 3 & 0 \\ 3 & 1 & 1 & 0 & 1 \\ 0 & 3 & 0 & 0 & 2 \\ 4 & 0 & 1 & 2 & 3 \end{bmatrix}$

In Exercises 19–21 find the adjacency matrix of the given directed multigraph with respect to the vertices listed in alphabetic order.

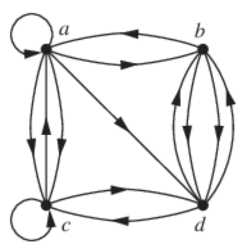
**19.**




**20.**




**21.**




In Exercises 22–24 draw the graph represented by the given adjacency matrix.

**22.** $\begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ **23.** $\begin{bmatrix} 1 & 2 & 1 \\ 2 & 0 & 0 \\ 0 & 2 & 2 \end{bmatrix}$ **24.** $\begin{bmatrix} 0 & 2 & 3 & 0 \\ 1 & 2 & 2 & 1 \\ 2 & 1 & 1 & 0 \\ 1 & 0 & 0 & 2 \end{bmatrix}$

**25.** Is every zero–one square matrix that is symmetric and has zeros on the diagonal the adjacency matrix of a simple graph?

**26.** Use an incidence matrix to represent the graphs in Exercises 1 and 2.

**27.** Use an incidence matrix to represent the graphs in Exercises 13–15.

***28.** What is the sum of the entries in a row of the adjacency matrix for an undirected graph? For a directed graph?

***29.** What is the sum of the entries in a column of the adjacency matrix for an undirected graph? For a directed graph?

**30.** What is the sum of the entries in a row of the incidence matrix for an undirected graph?

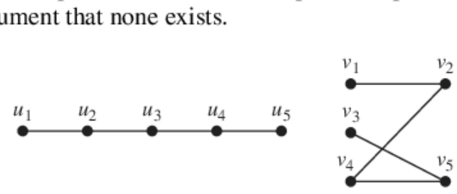
**31.** What is the sum of the entries in a column of the incidence matrix for an undirected graph?

***32.** Find an adjacency matrix for each of these graphs.
**a)** $K_n$ **b)** $C_n$ **c)** $W_n$ **d)** $K_{m,n}$ **e)** $Q_n$

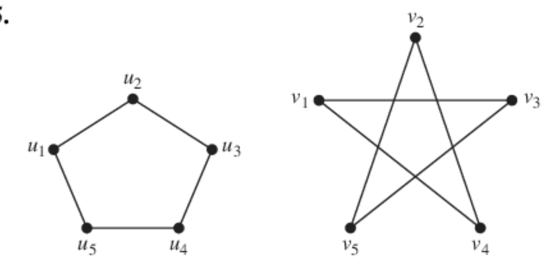
***33.** Find incidence matrices for the graphs in parts (a)–(d) of Exercise 32.

In Exercises 34–44 determine whether the given pair of graphs is isomorphic. Exhibit an isomorphism or provide a rigorous argument that none exists.

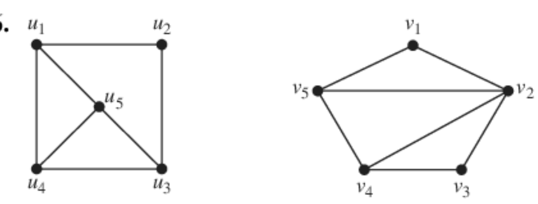
**34.**



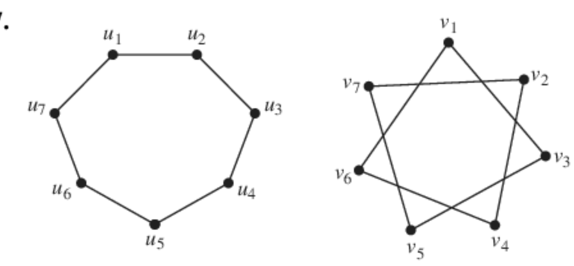

**35.**



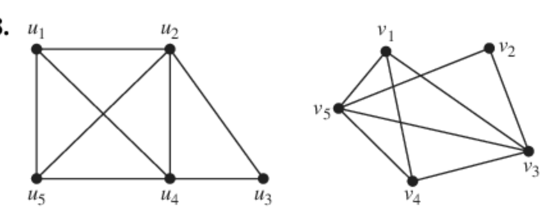

**36.**



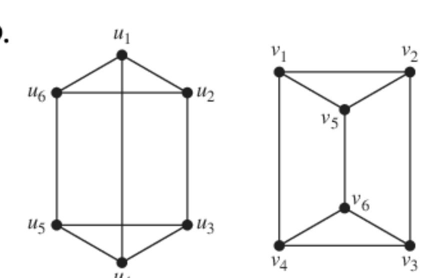

**37.**



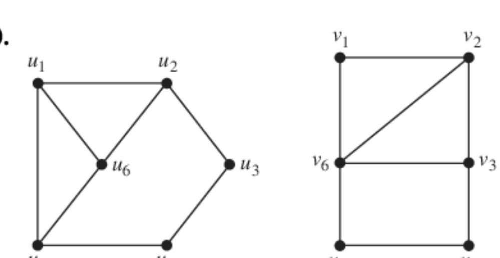

**38.**



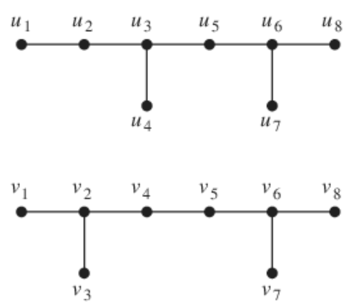

**39.**



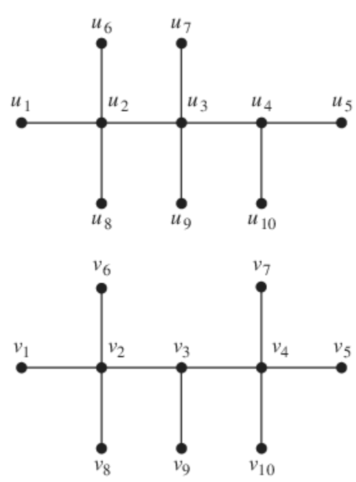

**40.**



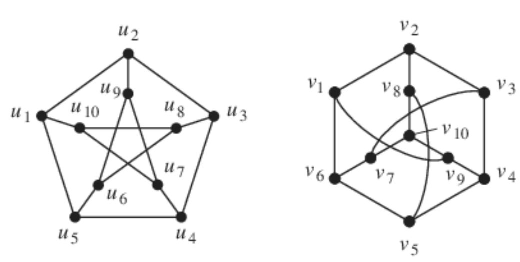
**41.**



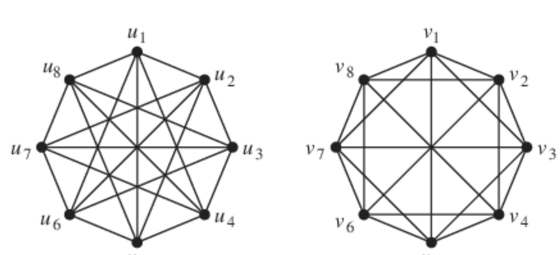

**42.**




**43.**




**44.**




**45.** Show that isomorphism of simple graphs is an equivalence relation.

**46.** Suppose that $G$ and $H$ are isomorphic simple graphs. Show that their complementary graphs $\overline{G}$ and $\overline{H}$ are also isomorphic.

**47.** Describe the row and column of an adjacency matrix of a graph corresponding to an isolated vertex.

**48.** Describe the row of an incidence matrix of a graph corresponding to an isolated vertex.

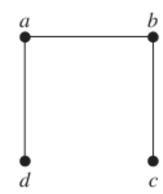
**49.** Show that the vertices of a bipartite graph with two or more vertices can be ordered so that its adjacency matrix has the form

$$\begin{bmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{B} & \mathbf{0} \end{bmatrix},$$

where the four entries shown are rectangular blocks.

A simple graph $G$ is called **self-complementary** if $G$ and $\overline{G}$ are isomorphic.

**50.** Show that this graph is self-complementary.




**51.** Find a self-complementary simple graph with five vertices.

***52.** Show that if $G$ is a self-complementary simple graph with $v$ vertices, then $v \equiv 0$ or $1 \pmod 4$.

**53.** For which integers $n$ is $C_n$ self-complementary?

**54.** How many nonisomorphic simple graphs are there with $n$ vertices, when $n$ is

**a)** 2? **b)** 3? **c)** 4?

**55.** How many nonisomorphic simple graphs are there with five vertices and three edges?

**56.** How many nonisomorphic simple graphs are there with six vertices and four edges?

**57.** Are the simple graphs with the following adjacency matrices isomorphic?

**a)** $\begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$

**b)** $\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$

**c)** $\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$

**58.** Determine whether the graphs without loops with these incidence matrices are isomorphic.

**a)** $\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$

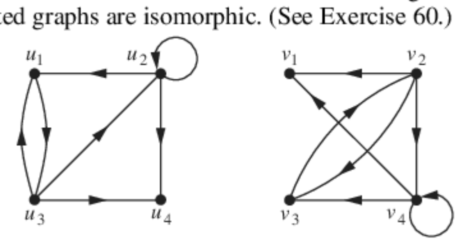
**b)** $\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$

**59.** Extend the definition of isomorphism of simple graphs to undirected graphs containing loops and multiple edges.

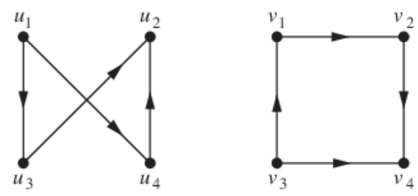
**60.** Define isomorphism of directed graphs.

In Exercises 61–64 determine whether the given pair of directed graphs are isomorphic. (See Exercise 60.)

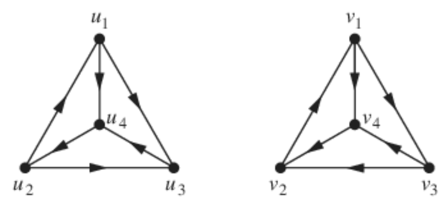
**61.**



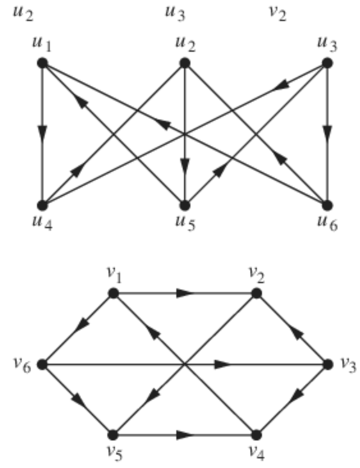

**62.**




**63.**




**64.**




**65.** Show that if $G$ and $H$ are isomorphic directed graphs, then the converses of $G$ and $H$ (defined in the preamble of Exercise 67 of Section 10.2) are also isomorphic.

**66.** Show that the property that a graph is bipartite is an isomorphic invariant.

**67.** Find a pair of nonisomorphic graphs with the same degree sequence (defined in the preamble to Exercise 36 in Section 10.2) such that one graph is bipartite, but the other graph is not bipartite.

***68.** How many nonisomorphic directed simple graphs are there with $n$ vertices, when $n$ is

**a)** 2? **b)** 3? **c)** 4?

***69.** What is the product of the incidence matrix and its transpose for an undirected graph?

***70.** How much storage is needed to represent a simple graph with $n$ vertices and $m$ edges using

**a)** adjacency lists?

**b)** an adjacency matrix?

**c)** an incidence matrix?

A **devil's pair** for a purported isomorphism test is a pair of nonisomorphic graphs that the test fails to show that they are not isomorphic.

**71.** Find a devil's pair for the test that checks the degree sequence (defined in the preamble to Exercise 36 in Section 10.2) in two graphs to make sure they agree.

**72.** Suppose that the function $f$ from $V_1$ to $V_2$ is an isomorphism of the graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Show that it is possible to verify this fact in time polynomial in terms of the number of vertices of the graph, in terms of the number of comparisons needed.

# 10.4 Connectivity

## Introduction

Many problems can be modeled with paths formed by traveling along the edges of graphs. For instance, the problem of determining whether a message can be sent between two computers using intermediate links can be studied with a graph model. Problems of efficiently planning routes for mail delivery, garbage pickup, diagnostics in computer networks, and so on can be solved using models that involve paths in graphs.

## Paths

Informally, 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. As the path travels along its edges, it visits the vertices along this path, that is, the endpoints of these edges.

A formal definition of paths and related terminology is given in Definition 1.

**DEFINITION 1** Let $n$ be a nonnegative integer and $G$ an undirected graph. A *path* of *length* $n$ from $u$ to $v$ 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 $x_0, x_1, \ldots, x_n$ (because listing these vertices uniquely determines the path). 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.

When it is not necessary to distinguish between multiple edges, we will denote a path $e_1, e_2, \ldots, e_n$, where $e_i$ is associated with $\{x_{i-1}, x_i\}$ for $i = 1, 2, \ldots, n$ by its vertex sequence $x_0, x_1, \ldots, x_n$. This notation identifies a path only as far as which vertices it passes through. Consequently, it does not specify a unique path when there is more than one path that passes through this sequence of vertices, which will happen if and only if there are multiple edges between some successive vertices in the list. Note that a path of length zero consists of a single vertex.

***Remark:*** There is considerable variation of terminology concerning the concepts defined in Definition 1. For instance, in some books, the term **walk** is used instead of *path*, where a walk is defined to be an alternating sequence of vertices and edges of a graph, $v_0, e_1, v_1, e_2, \ldots, v_{n-1}, e_n, v_n$, where $v_{i-1}$ and $v_i$ are the endpoints of $e_i$ for $i = 1, 2, \ldots, n$. When this terminology is used, **closed walk** is used instead of *circuit* to indicate a walk that begins and ends at the same vertex, and **trail** is used to denote a walk that has no repeated edge (replacing the term *simple path*). When this terminology is used, the terminology **path** is often used for a trail with no repeated vertices, conflicting with the terminology in Definition 1. Because of this variation in terminology, you will need to make sure which set of definitions are used in a particular book or article when you read about traversing edges of a graph. The text [GrYe06] is a good reference for the alternative terminology described in this remark.

**EXAMPLE 1** In the simple graph shown in Figure 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. However, $d, e, c, a$ is not a path, because $\{e, c\}$ is not an edge. Note that $b, c, f, e, b$ is a circuit of length 4 because $\{b, c\}$, $\{c, f\}$, $\{f, e\}$, and $\{e, b\}$ are edges, and this path begins and ends at $b$. The path $a, b, e, d, a, b$, which is of length 5, is not simple because it contains the edge $\{a, b\}$ twice. ◀

Paths and circuits in directed graphs were introduced in Chapter 9. We now provide more general definitions.

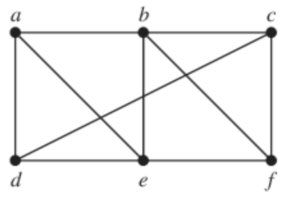



FIGURE 1 **A Simple Graph.**