1.3.14. Prove that every simple graph with at least two vertices has two vertices of equal degree. Is the conclusion true for loopless graphs?

1.3.15. For each $k \geq 3$, determine the smallest n such that

a) there is a simple k-regular graph with n vertices.

b) there exist nonisomorphic simple k-regular graphs with n vertices.

1.3.16. (+) For $k \geq 2$ and $g \geq 2$, prove that there exists an k-regular graph with girth g. (Hint: To construct such a graph inductively, make use of an $k-1$-regular graph H with girth g and a graph with girth $\lceil g/2 \rceil$ that is $n(H)$-regular. Comment: Such a graph with minimum order is a (k, g)-**cage**.) (Erdős–Sachs [1963])

1.3.17. (!) Let G be a graph with at least two vertices. Prove or disprove:

a) Deleting a vertex of degree $\Delta(G)$ cannot increase the average degree.

b) Deleting a vertex of degree $\delta(G)$ cannot reduce the average degree.

1.3.18. (!) For $k \geq 2$, prove that a k-regular bipartite graph has no cut-edge.

1.3.19. Let G be a claw-free graph. Prove that if $\Delta(G) \geq 5$, then G has a 4-cycle. For all $n \in \mathbb{N}$, construct a 4-regular claw-free graph of order at least n that has no 4-cycle.

1.3.20. (!) Count the cycles of length n in K_n and the cycles of length $2n$ in $K_{n,n}$.

1.3.21. Count the 6-cycles in $K_{m,n}$.

1.3.22. (!) Let G be a nonbipartite graph with n vertices and minimum degree k. Let l be the minimum length of an odd cycle in G.

a) Let C be a cycle of length l in G. Prove that every vertex not in $V(C)$ has at most two neighbors in $V(C)$.

b) By counting the edges joining $V(C)$ and $G - V(C)$ in two ways, prove that $n \geq kl/2$ (and thus $l \leq 2n/k$). (Campbell–Staton [1991])

c) When k is even, prove that the inequality of part (b) is best possible. (Hint: form a graph having $k/2$ pairwise disjoint l-cycles.)

1.3.23. Use the recursive description of Q_k (Example 1.3.8) to prove that $e(Q_k) = k2^{k-1}$.

1.3.24. Prove that $K_{2,3}$ is not contained in any hypercube Q_k.

1.3.25. (!) Prove that every cycle of length $2r$ in a hypercube is contained in a subcube of dimension at most r. Can a cycle of length $2r$ be contained in a subcube of dimension less than r?

1.3.26. (!) Count the 6-cycles in Q_3. Prove that every 6-cycle in Q_k lies in exactly one 3-dimensional subcube. Use this to count the 6-cycles in Q_k for $k \geq 3$.

1.3.27. Given $k \in \mathbb{N}$, let G be the subgraph of Q_{2k+1} induced by the vertices in which the number of ones and zeros differs by 1. Prove that G is regular, and compute $n(G)$, $e(G)$, and the girth of G.

1.3.28. Let V be the set of binary k-tuples. Define a simple graph Q'_k with vertex set V by putting $u \leftrightarrow v$ if and only if u and v *agree* in exactly one coordinate. Prove that Q'_k is isomorphic to the hypercube Q_k if and only if k is even. (D.G. Hoffman)

1.3.29. (∗+) *Automorphisms of the k-dimensional cube Q_k.*

a) Prove that every copy of Q_j in Q_k is a subgraph induced by a set of 2^j vertices having specified values on a fixed set of $k - j$ coordinates. (Hint: Prove that a copy of Q_j must have two vertices differing in j coordinates.)

b) Use part (a) to count the automorphisms of Q_k.

1.3.30. Prove that every edge in the Petersen graph belongs to exactly four 5-cycles, and use this to show that the Petersen graph has exactly twelve 5-cycles. (Hint: For the first part, extend the edge to a copy of P_4 and apply Proposition 1.1.38.)

1.3.31. (!) Use complete graphs and counting arguments (not algebra!) to prove that
a) $\binom{n}{2} = \binom{k}{2} + k(n-k) + \binom{n-k}{2}$ for $0 \le k \le n$.
b) If $\sum n_i = n$, then $\sum \binom{n_i}{2} \le \binom{n}{2}$.

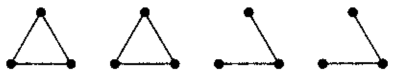
1.3.32. (!) Prove that the number of simple even graphs with vertex set $[n]$ is $2^{\binom{n-1}{2}}$. (Hint: Establish a bijection to the set of all simple graphs with vertex set $[n-1]$.)

1.3.33. (+) Let G be a triangle-free simple n-vertex graph such that every pair of nonadjacent vertices has exactly two common neighbors.
a) Prove that $n(G) = 1 + \binom{d(x)}{2}$, where $x \in V(G)$. Conclude that G is regular.
b) When $k = 5$, prove that deleting any one vertex and its neighbors from G leaves the Petersen graph. (Comment: When $k = 5$, the graph G is in fact the graph obtained from Q_4 by adding edges joining complementary vertices.)

1.3.34. (+) Let G be a kite-free simple n-vertex graph such that every pair of nonadjacent vertices has exactly two common neighbors. Prove that G is regular. (Galvin)

1.3.35. (+) Let n and k be integers such that $1 < k < n-1$. Let G be a simple n-vertex graph such that every k-vertex induced subgraph of G has m edges.
a) Let G' be an induced subgraph of G with l vertices, where $l > k$. Prove that $e(G') = m\binom{l}{k}/\binom{l-2}{k-2}$.
b) Use part (a) to prove that $G \in \{K_n, \overline{K}_n\}$. (Hint: Use part (a) to prove that the number of edges with endpoints u, v is independent of the choice of u and v.)

1.3.36. Let G be a 4-vertex graph whose list of subgraphs obtained by deleting one vertex appears below. Determine G.



1.3.37. Let H be a graph formed by deleting a vertex from a loopless regular graph G with $n(G) \ge 3$. Describe (and justify) a method for obtaining G from H.

1.3.38. Let G be a graph with at least 3 vertices. Prove that G is connected if and only if at least two of the subgraphs obtained by deleting one vertex of G are connected. (Hint: Use Proposition 1.2.29.)

1.3.39. (*+) Prove that every disconnected graph G with at least three vertices is reconstructible. (Hint: Having used Exercise 1.3.38 to determine that G is disconnected, use $G_1, \ldots, G_n$ to find a component M of G that occurs the most times among the components with the maximum number of vertices, use Proposition 1.2.29 to choose v so that $L = M - v$ is connected, and reconstruct G by finding some $G - v_i$ in which a copy of M became a copy of L.)

1.3.40. (!) Let G be an n-vertex simple graph, where $n \ge 2$. Determine the maximum possible number of edges in G under each of the following conditions.
a) G has an independent set of size a.
b) G has exactly k components.
c) G is disconnected.

1.3.41. (!) Prove or disprove: If G is an n-vertex simple graph with maximum degree $\lceil n/2 \rceil$ and minimum degree $\lfloor n/2 \rfloor - 1$, then G is connected.

1.3.42. Let S be a set of vertices in a k-regular graph G such that no two vertices in S are adjacent or have a common neighbor. Use the pigeonhole principle to prove that $|S| \le \lfloor n(G)/(k+1) \rfloor$. Show that the bound is best possible for the cube Q_3. (Comment: The bound is not best possible for Q_4.)

1.3.43. (+) Let G be a simple graph with no isolated vertices, and let $a = 2e(G)/n(G)$ be the average degree in G. Let $t(v)$ denote the average of the degrees of the neighbors of v. Prove that $t(v) \ge a$ for some $v \in V(G)$. Construct an infinite family of connected graphs such that $t(v) > a$ for every vertex v. (Hint: For the first part, compute the average of $t(v)$, using that $x/y + y/x \ge 2$ when $x, y > 0$.) (Ajtai–Komlós–Szemerédi [1980])

1.3.44. (!) Let G be a loopless graph with average vertex degree $a = 2e(G)/n(G)$.

a) Prove that $G - x$ has average degree at least a if and only if $d(x) \le a/2$.

b) Use part (a) to give an algorithmic proof that if $a > 0$, then G has a subgraph with minimum degree greater than $a/2$.

c) Show that there is no constant c greater than $1/2$ such that G must have a subgraph with minimum degree greater than ca; this proves that the bound in part (b) is best possible. (Hint: Use $K_{1,n-1}$.)

1.3.45. Determine the maximum number of edges in a bipartite subgraph of the Petersen graph.

1.3.46. Prove or disprove: Whenever the algorithm of Theorem 1.3.19 is applied to a bipartite graph, it finds the bipartite subgraph with the most edges (the full graph).

1.3.47. Use induction on $n(G)$ to prove that every nontrivial loopless graph G has a bipartite subgraph H such that H has *more* than $e(G)/2$ edges.

1.3.48. Construct graphs $G_1, G_2, \ldots$, with G_n having $2n$ vertices, such that $\lim_{n\to\infty} f_n = 1/2$, where f_n is the fraction of $E(G_n)$ belonging to the largest bipartite subgraph of G_n.

1.3.49. For each $k \in \mathbb{N}$ and each loopless graph G, prove that G has a k-partite subgraph H (Definition 1.1.12) such that $e(H) \ge (1 - 1/k)e(G)$.

1.3.50. (+) For $n \ge 3$, determine the minimum number of edges in a connected n-vertex graph in which every edge belongs to a triangle. (Erdős [1988])

1.3.51. (+) Let G be a simple n-vertex graph, where $n > 3$.

a) Use Proposition 1.3.11 to prove that if G has more than $n^2/4$ edges, then G has a vertex whose deletion leaves a graph with more than $(n-1)^2/4$ edges. (Hint: In every graph, the number of edges is an integer.)

b) Use part (a) to prove by induction that G contains a triangle if $e(G) > n^2/4$.

1.3.52. Prove that every n-vertex triangle-free simple graph with the maximum number of edges is isomorphic to $K_{\lfloor n/2 \rfloor, \lceil n/2 \rceil}$. (Hint: Strengthen the proof of Theorem 1.3.23.)

1.3.53. (!) Each game of *bridge* involves two teams of two partners each. Consider a club in which four players cannot play a game if two of them have previously been partners that night. Suppose that 15 members arrive, but one decides to study graph theory. The other 14 people play until each has been a partner with four others. Next they succeed in playing six more games (12 partnerships), but after that they cannot find four players containing no pair of previous partners. Prove that if they can convince the graph theorist to play, then at least one more game can be played. (Adapted from Bondy–Murty [1976, p111]).

1.3.54. (+) Let G be a simple graph with n vertices. Let $t(G)$ be the total number of triangles in G and $\overline{G}$ together.

a) Prove that $t(G) = \binom{n}{3} - (n-2)e(G) + \sum_{v\in V(G)} \binom{d(v)}{2}$ triangles. (Hint: Consider the contribution made to each side by each triple of vertices.

b) Prove that $t(G) \ge n(n-1)(n-5)/24$. (Hint: Use a lower bound on $\sum_{v\in V(G)} \binom{d(v)}{2}$ in terms of average degree.)

c) When $n-1$ is divisible by 4, construct a graph achieving equality in part (b). (Goodman [1959])

1.3.55. (+) *Maximum size with no induced P_4.*

a) Let G be the complement of a disconnected simple graph. Prove that $e(G) \le \Delta(G)^2$, with equality only for $K_{\Delta(G),\Delta(G)}$.

b) Let G be a simple connected P_4-free graph with maximum degree k. Prove that $e(G) \le k^2$. (Seinsche [1974], Chung–West [1993])

1.3.56. Use induction (on n or on $\sum d_i$) to prove that if $d_1, \dots, d_n$ are nonnegative integers and $\sum d_i$ is even, then there is an n-vertex graph with vertex degrees $d_1, \dots, d_n$. (Comment: This requests an alternative proof of Proposition 1.3.28.)

1.3.57. (!) Let n be a positive integer. Let d be a list of n nonnegative integers with even sum whose largest entry is less than n and differs from the smallest entry by at most 1. Prove that d is graphic. (Hint: Use the Havel–Hakimi Theorem. Example: 443333 is such a list, as is 33333322.)

1.3.58. *Generalization of Havel–Hakimi Theorem.* Given a nonincreasing list d of nonnegative integers, let d' be obtained by deleting d_k and subtracting 1 from the k largest elements remaining in the list. Prove that d is graphic if and only if d' is graphic. (Hint: Mimic the proof of Theorem 1.3.31.) (Wang–Kleitman [1973])

1.3.59. Define $d = (d_1, \dots, d_{2k})$ by $d_{2i} = d_{2i-1} = i$ for $1 \le i \le k$. Prove that d is graphic. (Hint: Do not use the Havel–Hakimi Theorem.)

1.3.60. (+) Let d be a list of integers consisting of k copies of a and $n-k$ copies of b, with $a \ge b \ge 0$. Determine necessary and sufficient conditions for d to be graphic.

1.3.61. (!) Suppose that $G \cong \overline{G}$ and that $n(G) \equiv 1 \bmod 4$. Prove that G has at least one vertex of degree $(n(G)-1)/2$.

1.3.62. Suppose that n is congruent to 0 or 1 modulo 4. Construct an n-vertex simple graph G with $\frac{1}{2}\binom{n}{2}$ edges such that $\Delta(G) - \delta(G) \le 1$.

1.3.63. (!) Let $d_1, \dots, d_n$ be integers such that $d_1 \ge \cdots \ge d_n \ge 0$. Prove that there is a loopless graph (multiple edges allowed) with degree sequence $d_1, \dots, d_n$ if and only if $\sum d_i$ is even and $d_1 \le d_2 + \cdots + d_n$. (Hakimi [1962])

1.3.64. (!) Let $d_1 \le \cdots \le d_n$ be the vertex degrees of a simple graph G. Prove that G is connected if $d_j \ge j$ when $j \le n-1-d_n$. (Hint: Consider a component that omits some vertex of maximum degree.)

1.3.65. (+) Let $a_1 < \cdots < a_k$ be distinct positive integers. Prove that there is a simple graph with $a_k + 1$ vertices whose set of distinct vertex degrees is $a_1, \dots, a_k$. (Hint: Use induction on k to construct such a graph.) (Kapoor–Polimeni–Wall [1977])

1.3.66. (∗) *Expansion of 3-regular graphs* (see Example 1.3.26). For $n = 4k$, where $k \ge 2$, construct a connected 3-regular simple graph with n vertices that has no cut-edge but cannot be obtained from a smaller 3-regular simple graph by expansion. (Hint:

The desired graph must have no edge to which the inverse "erasure" operation can be applied to obtain a smaller simple graph.)

1.3.67. $(*)$ *Construction of 3-regular simple graphs*

a) Prove that a 2-switch can be performed by performing a sequence of expansions and erasures; these operations are defined in Example 1.3.26. (Caution: Erasure is not allowed when it would produce multiple edges.)

b) Use part (a) to prove that every 3-regular simple graph can be obtained from K_4 by a sequence of expansions and erasures. (Batagelj [1984])

1.3.68. $(*)$ Let G and H be two simple bipartite graphs, each with bipartition X, Y. Prove that $d_G(v) = d_H(v)$ for all $v \in X \cup Y$ if and only if there is a sequence of 2-switches that transforms G into H without ever changing the bipartition (each 2-switch replaces two edges joining X and Y by two other edges joining X and Y).

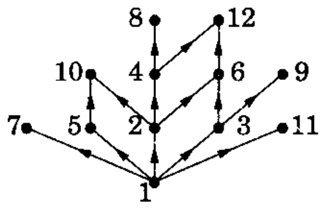
1.4. Directed Graphs

We have used graphs to model symmetric relations. Relation need not be symmetric; in general, a relation on S can be any set of ordered pairs in $S \times S$ (see Appendix A). For such relations, we need a more general model.

DEFINITIONS AND EXAMPLES

Seeking a graphical representation of the information in a general relation on S leads us to a model of directed graphs.

1.4.1. Example. For natural numbers x, y, we say that x is a "maximal divisor" of y if y/x is a prime number. For $S \subseteq \mathbb{N}$, the set $R = \{(x, y) \in S^2 : x \text{ is a maximal divisor of } y\}$ is a relation on S. To represent it graphically, we name a point in the plane for each element of S and draw an arrow from x to y whenever $(x, y) \in R$. Below we show the result when $S = [12]$. ■



1.4.2. Definition. A **directed graph** or **digraph** G is a triple consisting of a **vertex set** $V(G)$, an **edge set** $E(G)$, and a function assigning each edge an ordered pair of vertices. The first vertex of the ordered pair is the **tail** of the edge, and the second is the **head**; together, they are the **endpoints**. We say that an edge is an edge **from** its tail **to** its head.

The terms "head" and "tail" come from the arrows used to draw digraphs. As with graphs, we assign each vertex a point in the plane and each edge a curve joining its endpoints. When drawing a digraph, we give the curve a direction from the tail to the head.

When a digraph models a relation, each ordered pair is the (head, tail) pair for at most one edge. In this setting as with simple graphs, we ignore the technicality of a function assigning endpoints to edges and simply treat an edge as an ordered pair of vertices.

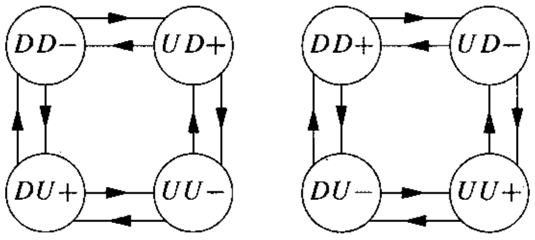
1.4.3. Definition. In a digraph, a **loop** is an edge whose endpoints are equal. **Multiple edges** are edges having the same ordered pair of endpoints. A digraph is **simple** if each ordered pair is the head and tail of at most one edge; one loop may be present at each vertex.

In a simple digraph, we write uv for an edge with tail u and head v. If there is an edge from u to v, then v is a **successor** of u, and u is a **predecessor** of v. We write $u \to v$ for "there is an edge from u to v".

1.4.4. Application. A **finite state machine** (also called **finite automaton** or **discrete system**) has a number of possible "states". Such a system can be modeled using a digraph in which vertices represent the states and edges represent the possible transitions between states.

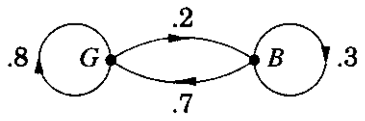
Transitions inherently move in one direction, so digraphs provide the appropriate model. Labels on the edges can be used to record the events that cause the transitions. When an event causes the system to remain in the same state, we have a loop. When two types of events can cause a particular transition, we might use multiple edges.

Consider a light controlled by two switches, often called a "three-way switch". The first switch can be up or down, the second switch can be up or down, and the light can be on (+) or off (−). Thus there are eight states. Transitions between states result by flipping switches. In the drawing below, the horizontal edges represent transitions caused by flipping the first switch, and the vertical edges represent transitions caused by flipping the second switch. (Drawing vertices large enough to put labels inside is not uncommon when discussing finite state machines, but we will stick with filled dots.) ■



1.4.5.* Application. Edge labels can be used to record transition probabilities when a system operates randomly. The probabilities on the edges leaving a vertex sum to 1, and the system is called a **Markov chain**. Methods of linear algebra can be used to compute the long-term fraction of time spent in each state.

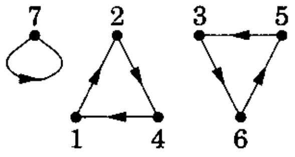
For example, suppose that weather has two states: good and bad. Air masses move slowly enough that tomorrow's weather tends to be like today's. In most places, storm systems don't linger long, so we might have transition probabilities as shown below. If we record states hourly instead of daily, then the probably of remaining in the same state is much higher. ■



1.4.6. Definition. A digraph is a **path** if it is a simple digraph whose vertices can be linearly ordered so that there is an edge with tail u and head v if and only if v immediately follows u in the vertex ordering. A **cycle** is defined similarly using an ordering of the vertices on a circle.

1.4.7. Example. *Functional digraphs.* We can study a function $f\colon A \to A$ using digraphs. The **functional digraph** of f is the simple digraph with vertex set A and edge set $\{(x, f(x)\colon x \in A\}$. For each x, the single edge with tail x points to the image of x under f.

Following a path in a functional digraph corresponds to iterating the function. In a permutation, each element is the image of exactly one element, so the functional digraph has one head and one tail at each vertex. Hence the functional digraph of a permutation consists of disjoint cycles. Below we show the functional digraph for a permutation of [7]. ■

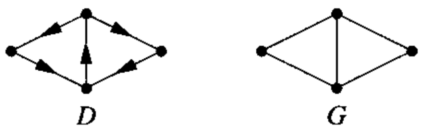


1.4.8.* Remark. We often use the same names for corresponding concepts in the graph and digraph models. Many authors replace "vertex" and "edge" with "node" and "arc" to discuss digraphs, but this obscures the analogies. Some results have the same statements and proofs; it would be wasteful to repeat them just to change terminology (especially in Chapter 4).

Also, a graph G can be modeled using a digraph D in which each edge $uv \in E(G)$ is replaced with $uv, vu \in E(D)$. In this way, results about digraphs can be applied to graphs. Since the notion of "edge" in digraphs extends the notion of "edge" in graphs, using the same name makes sense.

Some authors write "directed path" and "directed cycle" for our concepts of path and cycle in digraphs, but the distinction is unnecessary; for the "weak" version that does not follow the arrows, we can speak of a path or cycle in the graph obtained by ignoring the directions, which we define next. ■

1.4.9. Definition. The **underlying graph** of a digraph D is the graph G obtained by treating the edges of D as unordered pairs; the vertex set and edge set remain the same, and the endpoints of an edge are the same in G as in D, but in G they become an unordered pair.



Most ideas and methods of graph theory arise in the study of ordinary graphs. Digraphs can be a useful additional tool, especially in applications, as we have tried to suggest. We hope that describing the analogies and contrasts between graphs and digraphs will help clarify the concepts.

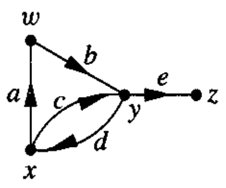
When comparing a digraph with a graph, we usually use G for the graph and D for the digraph. When discussing a single digraph, we often use G.

1.4.10. Definition. The definitions of **subgraph, isomorphism, decomposition**, and **union** are the same for graphs and digraphs. In the **adjacency matrix** $A(G)$ of a digraph G, the entry in position i, j is the number of edges from v_i to v_j. In the **incidence matrix** $M(G)$ of a loopless digraph G, we set $m_{i,j} = +1$ if v_i is the tail of e_j and $m_{i,j} = -1$ if v_i is the head of e_j.

1.4.11. Example. The underlying graph of the digraph below is the graph of Example 1.1.19; note the similarities and differences in their matrices. ■

$$\begin{array}{c|cccc} & w & x & y & z \\ \hline w & 0 & 0 & 1 & 0 \\ x & 1 & 0 & 1 & 0 \\ y & 0 & 1 & 0 & 1 \\ z & 0 & 0 & 0 & 0 \end{array}$$

$A(G)$



G

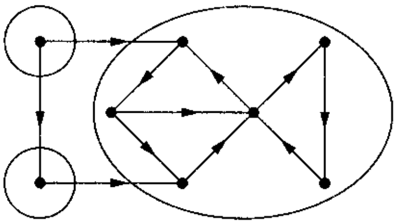
$$\begin{array}{c|ccccc} & a & b & c & d & e \\ \hline w & -1 & +1 & 0 & 0 & 0 \\ x & +1 & 0 & +1 & -1 & 0 \\ y & 0 & -1 & -1 & +1 & +1 \\ z & 0 & 0 & 0 & 0 & -1 \end{array}$$

$M(G)$

To define connected digraphs, two options come to mind. We could require only that the underlying graph be connected. However, this does not capture the most useful sense of connection for digraphs.

1.4.12. Definition. A digraph is **weakly connected** if its underlying graph is connected. A digraph is **strongly connected** or **strong** if for each *ordered pair* u, v of vertices, there is a path from u to v. The **strong components** of a digraph are its maximal strong subgraphs.

1.4.13. Example. The 2-vertex digraph consisting only of the edge xy has an x, y-path but no y, x-path and is not strongly connected. As a digraph, an n-vertex path has n strong components, but a cycle has only one. In the digraph below, the three circled subdigraphs are the strong components. Properties of strong components are discussed in Exercises 10–13. ■



1.4.14.* Application. *Games.* Many games with two players can be described as finite state machines. The vertex set is the set of possible states of the game. There is an edge from state x to state y if some move can be made (by the player whose turn it is to play) to reach state y from state x.

Let W be the set of vertices for winning positions; a player who brings the game to such a state wins. No edges leave W. A player who brings the game to a state with an edge to W loses, since the other player then reaches W. One way to analyze the game is to seek a set S of pairwise nonadjacent vertices containing W such that every vertex outside S has an edge to a vertex in S. A player who can bring the game to a position in S wins, but one who must move from a position in S loses.

For example, consider a game with two piles of pennies. At his or her turn, each player can remove any portion of a single pile. The player who removes the last coin wins. The possible game positions are the nonnegative integer pairs (r, s). The definition of the game specifies $(0, 0)$ as the only winning position. However, the set S of desirable positions is $\{(r, r): r \geq 0\}$. Since only one coordinate can decrease on a move, there is no edge within S. For each vertex $(r, s) \notin S$, a player can remove $|r - s|$ from the larger pile to reach S.

The general game of Nim starts with an arbitrary number of piles with arbitrary sizes, but otherwise the rules of the game are the same as this. Exercise 18 guarantees that Nim always has a winning strategy set S, since the digraph for this game has no cycles. If the initial position is in S, then the second player wins (assuming optimal play). Otherwise, the first player wins. ■

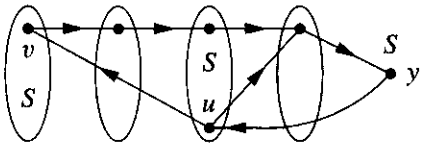
1.4.15.* Definition. A **kernel** in the digraph D is a set $S \subseteq V(D)$ such that S induces no edges and every vertex outside S has a successor in S.

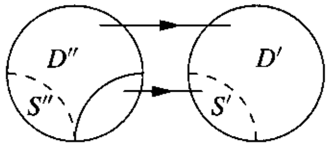
A digraph that is an odd cycle has no kernel (Exercise 17), but forbidding odd cycles as subdigraphs always yields a kernel. In proving this, all uses of paths, cycles, and walks are in the directed sense. We need several statements about movement in digraphs that hold by the same proofs as in graphs. For example, every u, v-walk in a digraph contains a u, v-path (Exercise 3), and every closed odd walk in a digraph contains an odd cycle (Exercise 4). The concept of **distance** from x to y will be explored more fully in Section 2.1; it is the least length of an x, y-path.

1.4.16.* Theorem. (Richardson [1953]) Every digraph having no odd cycle has a kernel.

Proof: Let D be such a digraph. We first consider the case that D is strongly connected; see the figure on the left below. Given an arbitrary vertex $y \in V(D)$, let S be the set of vertices with even distance to y. Every vertex with odd distance to y has a successor in S, as desired.

If the vertices of S are not pairwise nonadjacent, then there is an edge uv with $u, v \in S$. By the definition of S, there is a u, y-path P of even length and a v, y-path P' of even length. Adding uv at the start of P' yields a u, y-walk W of odd length. Because D is strong, D has a y, u-path Q. Combining Q with one of P or W yields a closed odd walk in D. This is impossible, since a closed odd walk contains an odd cycle. Thus S is a kernel in D.





For the general case, we use induction on $n(D)$.

Basis step: $n(D) = 1$. The only example is a single vertex with no loop. This vertex is a kernel by itself.

Induction step: $n(D) > 1$. Since we have already proved the claim for strong digraphs, we may assume that D is not strong. For some strong component D' of D, there is no edge from a vertex of D' to a vertex not in D' (Exercise 11). We have shown that D' has a kernel; let S' be a kernel of D'.

Let D'' be the subdigraph obtained from D by deleting D' and all the predecessors of S'. By the induction hypothesis, D'' has a kernel; let S'' be a kernel of D''. We claim that $S' \cup S''$ is a kernel of D. Since D'' has no predecessor of S', there is no edge within $S' \cup S''$. Every vertex in $D'' - S''$ has a successor in S'', and all other vertices not in $S' \cup S''$ have a successor in S'. ■

VERTEX DEGREES

In a digraph, we use the same notation for number of vertices and number of edges as in graphs. The notation for vertex degrees incorporates the distinction between heads and tails of edges.

1.4.17. Definition. Let v be a vertex in a digraph. The **outdegree** $d^+(v)$ is the number of edges with tail v. The **indegree** $d^-(v)$ is the number of edges with head v. The **out-neighborhood** or **successor set** $N^+(v)$ is $\{x \in V(G): v \to x\}$. The **in-neighborhood** or **predecessor set** $N^-(v)$ is $\{x \in V(G): x \to v\}$. The minimum and maximum indegree are $\delta^-(G)$ and $\Delta^-(G)$; for outdegree we use $\delta^+(G)$ and $\Delta^+(G)$.

The digraph analogue of the degree-sum formula for graphs is easy.

1.4.18. Proposition. In a digraph G, $\sum_{v\in V(G)} d^+(v) = e(G) = \sum_{v\in V(G)} d^-(v)$.

Proof: Every edge has exactly one tail and exactly one head. ∎

The digraph analogue of degree sequence is the list of "degree pairs" $(d^+(v_i), d^-(v_i))$. When is a list of pairs realizable as the degree pairs of a digraph? As with graphs, this is easy when we allow multiple edges.

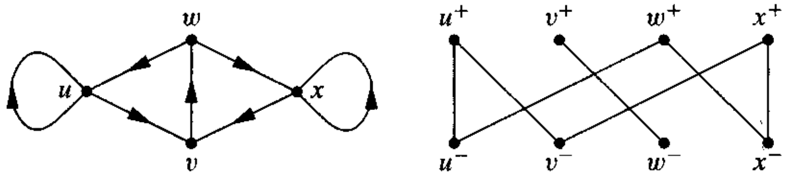
1.4.19.* Proposition. A list of pairs of nonnegative integers is realizable as the degree pairs of a digraph if and only if the sum of the first coordinates equals the sum of the second coordinates.

Proof: The condition is necessary because every edge has one tail and one head, contributing once to each sum.

For sufficiency, consider the pairs $\{(d_i^+, d_i^-): 1 \le i \le n\}$ and vertices $v_1, \dots, v_n$. Let $m = \sum d_i^+ = \sum d_j^-$. Consider m dots. Give the dots positive labels, with d_i^+ of them having label i. Also give the dots negative labels, with d_j^- of them having label $-j$. For each dot with labels i and $-j$, place an edge from v_i to v_j. This creates a digraph with $d^+(v_i) = d_i^+$ and $d^-(v_i) = d_i^-$. ∎

The analogous question for simple digraphs is harder. The question can be rephrased in terms of bipartite graphs via a transformation that is useful in many problems about digraphs.

1.4.20.* Definition. The **split** of a digraph D is a bipartite graph G whose partite sets V^+, V^- are copies of $V(D)$. For each vertex $x \in V(D)$, there is one vertex $x^+ \in V^+$ and one vertex $x^- \in V^-$. For each edge from u to v in D, there is an edge with endpoints u^+, v^- in G.



1.4.21.* Remark. The degrees of the vertices in the split of D are the indegrees and outdegrees of the vertices in D.

Furthermore, an X,Y-bigraph G with $|X| = |Y| = n$ can be transformed into an n-vertex digraph D by putting an edge $v_i v_j$ in D for each edge $x_i y_j$ in G; now G is the split of D. (This is one reason to allow loops in simple digraphs.)

Thus there is a simple digraph with degree pairs $\{(d_i^+, d_i^-): 1 \le i \le n\}$ if and only if there is a simple bipartite graph G in which the vertex degrees are $d_1^+, \dots, d_n^+$ in one partite set and $d_1^-, \dots, d_n^-$ in the other partite set. Exercise 32 obtains a recursive test for existence of such a bipartite graph. The statement and proof are like that of the Havel–Hakimi Theorem, so we leave further discussion to the exercise. ∎

EULERIAN DIGRAPHS

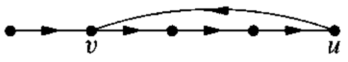
The definitions of **trail**, **walk**, **circuit**, and the **connection relation** are the same in graphs and digraphs when we list edges as ordered pairs of vertices. In a digraph, the successive edges must "follow the arrows". In a walk $v_0, e_1, \ldots, e_k, v_k$, the edge e_i has tail v_{i-1} and head v_i.

1.4.22. Definition. An **Eulerian trail** in a digraph (or graph) is a trail containing all edges. An **Eulerian circuit** is a closed trail containing all edges. A digraph is **Eulerian** if it has an Eulerian circuit.

The characterization of Eulerian digraphs is analogous to the characterization of Eulerian graphs. The proof is essentially the same as for graphs, so we leave it to the exercises.

1.4.23. Lemma. If G is a digraph with $\delta^+(G) \geq 1$, then G contains a cycle. The same conclusion holds when $\delta^-(G) \geq 1$.

Proof: Let P be a maximal path in G, and let u be the last vertex of P. Since P cannot be extended, every successor of u must already be a vertex of P. Since $\delta^+(G) \geq 1$, u has a successor v on P. The edge uv completes a cycle with the portion of P from v to u. ■



1.4.24. Theorem. A digraph is Eulerian if and only if $d^+(v) = d^-(v)$ for each vertex v and the underlying graph has at most one nontrivial component.

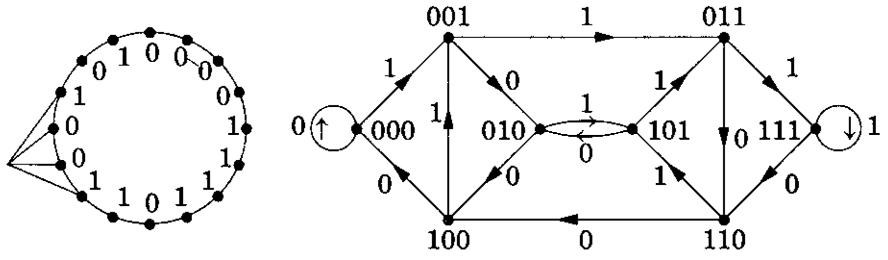
Proof: See Exercise 19 or Exercise 20. ■

Every Eulerian digraph with no isolated vertices is strongly connected, although the characterization states that being weakly connected is sufficient.

1.4.25. Application. *de Bruijn cycles.* There are 2^n binary strings of length n. Is there a cyclic arrangement of 2^n binary digits such that the 2^n strings of n consecutive digits are all distinct? For $n = 4$, (0000111101100101) works.

We can use such an arrangement to keep track of the position of a rotating drum (Good [1946]). Our drum has 2^n rotational positions. A band around the circumference is split into 2^n portions that can be coded 0 or 1. Sensors read n consecutive portions. If the coding has the property specified above, then the position of the drum is determined by the string read by the sensors.

To obtain such a circular arrangement, define a digraph D_n whose vertices are the binary $(n-1)$-tuples. Put an edge from a to b if the last $n-2$ entries of a agree with the first $n-2$ entries of b. Label the edge with the last entry of b. Below we show D_4. We next prove that D_n is Eulerian and show how an Eulerian circuit yields the desired circular arrangement. ■



1.4.26. Theorem. The digraph D_n of Application 1.4.25 is Eulerian, and the edge labels on the edges in any Eulerian circuit of D_n form a cyclic arrangement in which the 2^n consecutive segments of length n are distinct.

Proof: We show first that D_n is Eulerian. Every vertex has outdegree 2, because we can append a 0 or a 1 to its name to obtain the name of a successor vertex. Similarly, every vertex has indegree 2, because the same argument applies when moving in reverse and putting a 0 or a 1 on the front of the name. Also, D_n is strongly connected, because we can reach the vertex $b = (b_1, \ldots, b_{n-1})$ from any vertex by successively following the edges labeled $b_1, \ldots, b_{n-1}$. Thus D_n satisfies the hypotheses of Theorem 1.4.24 and is Eulerian.

Let C be an Eulerian circuit of D_n. Arrival at vertex $a = (a_1, \ldots, a_{n-1})$ must be along an edge with label a_{n-1}, because the label on an edge entering a vertex agrees with the last entry of the name of the vertex. Since we delete the front and shift the rest to obtain the rest of the name at the head, the successive earlier labels (looking backward) must have been $a_{n-2}, \ldots, a_1$ in order. If C next uses an edge with label a_n, then the list consisting of the n most recent edge labels at that time is $a_1, \ldots, a_n$.

Since the 2^{n-1} vertex labels are distinct, and the two edges leaving each vertex have distinct labels, and we traverse each edge from each vertex exactly once along C, we have shown that the 2^n strings of length n in the circular arrangement given by the edge labels along C are distinct. ■

The digraph D_n is the **de Bruijn graph** of order n on an alphabet of size 2. It is useful for other purposes, because it has many vertices and few edges (only twice the number of vertices) and yet we can reach each vertex from any other by a short path. We can reach any desired vertex in $n - 1$ steps by introducing the bits in its name in order from the current vertex.

ORIENTATIONS AND TOURNAMENTS

There are n^2 ordered pairs of elements that can be formed from a vertex set of size n. A simple digraph allows loops but uses each ordered pair at most once as an edge. Thus there are n^2 ordered pairs that may or may not be present as edges, and there are 2^{n^2} simple digraphs with vertex set $v_1, \ldots, v_n$.

Sometimes we want to forbid loops.

1.4.27. Definition. An **orientation** of a graph G is a digraph D obtained from G by choosing an orientation ($x \to y$ or $y \to x$) for each edge $xy \in E(G)$. An **oriented graph** is an orientation of a simple graph. A **tournament** is an orientation of a complete graph.

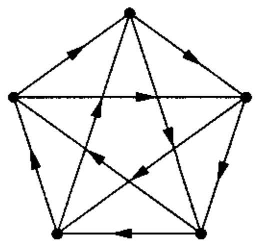
An oriented graph is the same thing as a loopless simple digraph. When the edges of a graph represent comparisons to be performed among items corresponding to the vertices, we can record the results by putting $x \to y$ when x does better than y in the comparison. The outcome is an orientation of G.

The number of oriented graphs with vertices $v_1, \dots, v_n$ is $3^{\binom{n}{2}}$; the number of tournaments is $2^{\binom{n}{2}}$.

1.4.28. Example. Orientations of complete graphs model "round-robin tournaments". Consider an n-team league where each team plays every other exactly once. For each pair u, v, we include the edge uv if u wins or vu if v wins. At the end of the season we have an orientation of K_n. The "score" of a team is its outdegree, which equals its number of wins.

We therefore call the outdegree sequence of a tournament its **score sequence.** The outdegrees determine the indegrees, since $d^+(v) + d^-(v) = n - 1$ for every vertex v. It is easier to characterize the score sequences of tournaments than the degree sequences of simple graphs (Exercise 35). ■

A tournament may have more than one vertex with maximum outdegree, so there may be no clear "winner"—in the example below, every vertex has outdegree 2 and indegree 2. Choosing a champion when several teams have the maximum number of wins can be difficult. Although there need not be a clear winner, we show next that there must always be a team x such that, for every other team z, either x beats z or x beats some team that beats z.

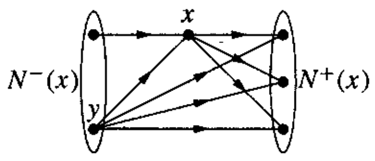


1.4.29. Definition. In a digraph, a **king** is a vertex from which every vertex is reachable by a path of length at most 2.

1.4.30. Proposition. (Landau [1953]) Every tournament has a king.

Proof: Let x be a vertex in a tournament T. If x is not a king, then some vertex y is not reachable from x by a path of length at most 2. Hence no successor of x is a predecessor of y. Since T is an orientation of a clique, every successor of x must therefore be a successor of y. Also $y \to x$. Hence $d^+(y) > d^+(x)$.

If y is not a king, then we repeat the argument to find z with yet larger outdegree. Since T is finite, we cannot forever obtain vertices of successively higher outdegree. The procedure must terminate, and it can terminate only when we have found a king. ■



In the language of extremality, we have proved that every vertex of maximum outdegree in a tournament is a king. Exercises 36–38 ask further questions about kings (see also Maurer [1980]). Exercise 39 generalizes the result to arbitrary digraphs.

EXERCISES

1.4.1. (−) Describe a relation in the real world whose digraph has no cycles. Describe another that has cycles but is not symmetric.

1.4.2. (−) In the lightswitch system of Application 1.4.4, suppose the first switch becomes disconnected from the wiring. Draw the digraph that models the resulting system.

1.4.3. (−) Prove that every u, v-walk in a digraph contains a u, v-path.

1.4.4. (−) Prove that every closed walk of odd length in a digraph contains the edges of an odd cycle. (Hint: Follow Lemma 1.2.15.)

1.4.5. (−) Let G be a digraph in which indegree equals outdegree at each vertex. Prove that G decomposes into cycles.

1.4.6. (−) Draw the de Bruijn graphs D_2 and D_4.

1.4.7. (−) Prove or disprove: If D is an orientation of a simple graph with 10 vertices, then the vertices of D cannot have distinct outdegrees.

1.4.8. (−) Prove that there is an n-vertex tournament with indegree equal to outdegree at every vertex if and only if n is odd.

• • • • •

1.4.9. For each $n \geq 1$, prove or disprove: Every simple digraph with n vertices has two vertices with the same outdegree or two vertices with the same indegree.

1.4.10. (!) Prove that a digraph is strongly connected if and only if for each partition of the vertex set into nonempty sets S and T, there is an edge from S to T.

1.4.11. (!) Prove that in every digraph, some strong component has no entering edges, and some strong component has no exiting edges.

1.4.12. Prove that in a digraph the connection relation is an equivalence relation, and its equivalence classes are the vertex sets of the strong components.