KARPAGAM ACADEMY OF HIGHER EDUCATION

(Deemed to be University Established Under Section 3 of UGC Act 1956)

Coimbatore – 641 021.

SYLLABUS

18MMP205A **GRAPH THEORY AND ITS APPLICATIONS** **4H – 4C**

Instruction Hours / week: L: 4 T: 0 P: 0 **Marks:** Internal: **40** External: **60** Total: **100**

End Semester Exam: 3 Hours

Course Objectives

This course enables the students to learn

- The fundamental concepts in Graph Theory and some of its modern applications.
- The use of these methods in subsequent courses in the design and analysis of algorithms, computability theory, software engineering, and computer systems.

Course Outcomes (COs)

1. Understanding the basic concepts of graphs, directed graphs, and weighted graphs and able to present a graph by matrices.
2. Overview of properties of trees and a minimal spanning tree for a given weighted graph.
3. Understand Eulerian and Hamiltonian graphs.
4. Applied the knowledge of graphs to solve the real-life problem.

UNIT I

GRAPHS

Graphs – Introduction – Isomorphism – Sub graphs – Walks, Paths, Circuits – Connectedness – Components – Euler Graphs – Hamiltonian Paths and Circuits – Trees – Properties of trees – Distance and Centers in Tree – Rooted and Binary Trees - Spanning trees – Fundamental Circuits.

UNIT II

SPANNING TREES

Spanning Trees in a Weighted Graph – Cut Sets – Properties of Cut Set – All Cut Sets – Fundamental Circuits and Cut Sets – Connectivity and separability – Network flows – 1-Isomorphism – 2-Isomorphism – Combinational versus Geometric Graphs – Planer Graphs – Different Representation of a Planer Graph.

UNIT III

MATRIX REPRESENTATION OF A GRAPH

Incidence matrix – Sub matrices – Circuit Matrix – Path Matrix – Adjacency Matrix – Chromatic Number – Chromatic partitioning – Chromatic polynomial - Matching - Covering – Four Color Problem.

UNIT IV

COUNTING TREE

Directed Graphs – Types of Directed Graphs - Types of enumeration, counting labeled trees, counting unlabelled trees, Polya's counting theorem, graph enumeration with Polya's theorem.

UNIT V

DOMINATION IN GRAPHS

Introduction – Terminology and concepts – Applications – Dominating set and domination number – Independent set and independence number – History of domination in graphs.

SUGGESTED READINGS

1. Deo N, (2007). Graph Theory with Applications to Engineering and Computer Science, Prentice Hall of India Pvt Ltd, New Delhi.
2. Teresa W. Haynes, Stephen T. Hedetniemi and Peter J.Slater, (1998), Fundamentals of Domination in Graphs, Marcel Dekker, New York.
3. Jonathan L Gross, Jay Yellen, (2014). Handbook of Graph Theory, CRC Press LLC. Taylor &Francis Group,Boca Rotan.
4. Diestel. R Springer-Verlag, (2012). Graph Theory. Springer-Verleg,New York.
5. Jensen.TR and Toft.B., (1995). Graph Coloring Problems. Wiley-Interscience , , New York.
6. Fred Buckley and Frank Harary, (1990). Distance in Graphs, Addison - Wesley Publications. Redwood City, California.
7. Flouds C. R., (2009). Graph Theory Applications, Narosa Publishing House. New Delhi,India.
8. Arumugam. S, Ramachandran. S ,(2006). Invitation to graph theory, Scitech publications, Chennai.
9. Harary F, (2001).Graph Theory, Addison- Wesley Publishing Company Inc USA

KARPAGAM ACADEMY OF HIGHER EDUCATION
(Deemed to be University Established Under Section 3 of UGC Act 1956)
Coimbatore – 641 021.

LECTURE PLAN
DEPARTMENT OF MATHEMATICS

Staff name: U. R. Ramakrishnan
Subject Name: Graph Theory and its Applications **Sub.Code: 18MMP205A**
Semester: II **Class: I M.Sc Mathematics**

S.No	Lecture Duration Period	Topics to be Covered	Support Material/Page Nos
UNIT-I			
1.	1	Isomorphism of graphs and sub graphs	R1:Chap:2.1:Pg.No:14- 16
2.	1	Walks, Paths, Circuits	R4:Chap:1.3:Pg.No:6-9
3.	1	Connected , connectedness of graphs and components of graphs	R1:Chap:2.5:Pg.No:19- 21
4.	1	Euler graphs and Euler graphs based on theorems	R1:Chap:2.6:Pg.No:21- 23
5.	1	Hamiltonian paths and circuits	R3:Chap:4.5:Pg.No:314-316
6.	1	Theorems on some properties of trees	R6:Chap:3:Pg.No:39-41
7.	1	Distance and centers in tree	R1:Chap:3.4:Pg.No:43- 45
8.	1	Rooted and binary trees and spanning trees, Fundamentals Circuits	R1:Chap:3.5:Pg.No:45- 57
9.	1	Recapitulation and Discussion of possible questions	
Total No of Hours Planned For Unit I=9			

R1: Deo N, (2007). Graph Theory with Applications to Engineering and Computer Science, Prentice Hall of India Pvt Ltd, New Delhi..
R3: Jonathan L Gross, Jay Yellen, (2014). Handbook of Graph Theory, CRC Press LLC. Taylor &Francis Group,Boca Rotan.
R4: Diestel. R Springer-Verlag, (2012). Graph Theory. Springer-Verleg,New York.
R6: Fred Buckley and Frank Harary, (1990). Distance in Graphs, Addison - Wesley Publications

Redwood City, California.			
UNIT-II			
1.	1	Spanning trees in a Weights Grap	R8:Chap:3.10:Pg.No:58- 61
2.	1	Theorems on some properties of Cut Sets and all Cut Sets	R1:Chap:4.2:Pg.No:68- 71
3.	1	Fundamental Circuits and Cut Sets	R1:Chap:4.5:Pg.No:73- 75
4.	1	Connectivity and separability	R1:Chap:4.5:Pg.No:73- 75
5.	1	Network flows	R3:Chap:11:Pg.No:1377-1380
6.	1	Theorems on some 1- Isomorphism	R1:Chap:4.7:Pg.No:80- 82
7.	1	Theorems on some 2- Isomorphism	R1:Chap:4.5:Pg.No:73- 75
8.	1	Combinational versus Geometric Graphs	R1:Chap:5.1:Pg.No:88- 89
9.	1	Different Representation of a Planar Graph	R1:Chap:5.4:Pg.No:90-99
10.	1	Recapitulation and Discussion of possible questions	
Total No of Hours Planned For Unit II=10			
R1: Deo N, (2007). Graph Theory with Applications to Engineering and Computer Science, Prentice Hall of India Pvt Ltd, New Delhi.. R3: Jonathan L Gross, Jay Yellen, (2014). Handbook of Graph Theory, CRC Press LLC. Taylor &Francis Group,Boca Rotan. R8: Arumugam. S, Ramachandran. S ,(2006). Invitation to graph theory, Scitech publications, Chennai.			
UNIT-III			
1.	1	Introduction and definition of a incidence matrix	R1:Chap:7.1:Pg.No:137-139
2.	1	Sub matrix and Circuits matrix based on problems	R1:Chap:7.3:Pg.No:142-146
3.	1	Path matrix and adjacency matrix based on problems	R1:Chap:7.8:Pg.No:156-161
4.	1	Chromatic Number theorems	R5:Chap:1.12:Pg.No:257 - 258
5.	1	Chromatic partitioning	R5:Chap:16.14:Pg.No:25 8-259
6.	1	Chromatic polynomial, Matching, covering	R1:Chap:8.5:Pg.No:174-190
7.	1	Four color problem	R5:Chap:2.1:Pg.No:31- 35

8.	1	Recapitulation and Discussion of possible questions	
Total No of Hours Planned For Unit III=8			
R1: Deo N, (2007). Graph Theory with Applications to Engineering and Computer Science, Prentice Hall of India Pvt Ltd, New Delhi.. R5: Jensen.TR and Toft.B., (1995). Graph Coloring Problems. Wiley-Interscience , , New York.			
UNIT-IV			
1.	1	Introduction and definition of Directed Graphs	R9:Chap:3.1:Pg.No:163-165
2.	1	Some types of Directed Graphs	R1:Chap:9.2:Pg.No:197-198
3.	1	Types of enumeration	R1:Chap:10.1:Pg.No:238 - 240
4.	1	Counting labeled trees	R1:Chap:10.2:Pg.No:240 - 241
5.	1	Counting unlabeled trees	R1:Chap:10.3:Pg.No:241 - 250
6.	1	Polya's counting theorem	R1:Chap:10.4:Pg.No:250 - 260
7.	1	Graph enumeration with Polya's theorem	R1:Chap:10.5:Pg.No:260 - 264
8.	1	Recapitulation and Discussion of possible questions	
Total No of Hours Planned For Unit IV=8			
R1: Deo N, (2007). Graph Theory with Applications to Engineering and Computer Science, Prentice Hall of India Pvt Ltd, New Delhi.. R9: Harary F, (2001).Graph Theory, Addison- Wesley Publishing Company Inc USA			
UNIT-V			
1.	1	Introduction Terminology and concepts	R1:Chap:1.1:Pg.No:15- 16
2.	1	Applications of Domination in graphs	R7:Chap:5.1:Pg.No:71-73
3.	1	Dominating set and Domination number	R2:Chap:1.2:Pg.No:16- 18
4.	1	Independent set and Independent number	R2:Chap:1.3:Pg.No:19- 20
5.	1	History of domination in graphs	R2:Chap:1.13:Pg.No:36- 37
6.	1	Recapitulation and Discussion of possible questions	
7.	1	Discuss on Previous ESE Question Papers	
8.	1	Discuss on Previous ESE Question Papers	

9.	1	Discuss on Previous ESE Question Papers	
Total No of Hours Planned for unit V=9			
R1: Deo N, (2007). Graph Theory with Applications to Engineering and Computer Science, Prentice Hall of India Pvt Ltd, New Delhi.. R2: Teresa W. Haynes, Stephen T. Hedetniemi and Peter J.Slater, (1998), Fundamentals of Domination in Graphs, Marcel Dekker, New York. R7: Flouds C. R., (2009). Graph Theory Applications, Narosa Publishing House. New Delhi,India.			
Total Planned Hours			44

UNIT-I

SYLLABUS

Graphs – Introduction – Isomorphism – Sub graphs – Walks, Paths, Circuits – Connectedness – Components – Euler Graphs – Hamiltonian Paths and Circuits – Trees – Properties of trees – Distance and Centers in Tree – Rooted and Binary Trees - Spanning trees – Fundamental Circuits

INTRODUCTION

A *linear*† *graph* (or simply a *graph*) $G = (V, E)$ consists of a set of objects $V = \{v_1, v_2, \ldots\}$ called *vertices*, and another set $E = \{e_1, e_2, \ldots\}$, whose elements are called *edges*, such that each edge e_k is identified with an unordered pair (v_i, v_j) of vertices. The vertices v_i, v_j associated with edge e_k are called the *end vertices* of e_k. The most common representation of a graph is by means of a diagram, in which the vertices are represented as points and each edge as a line segment joining its end vertices. Often this diagram itself is referred to as the graph. The object shown in Fig. 1-1, for instance, is a graph.

Observe that this definition permits an edge to be associated with a vertex pair (v_i, v_i). Such an edge having the same vertex as both its end vertices is called a *self-loop* (or simply a *loop*. The word loop, however, has a different meaning in electrical network theory; we shall therefore use the term self-loop to avoid confusion). Edge e_1 in Fig. 1-1 is a self-loop. Also note that

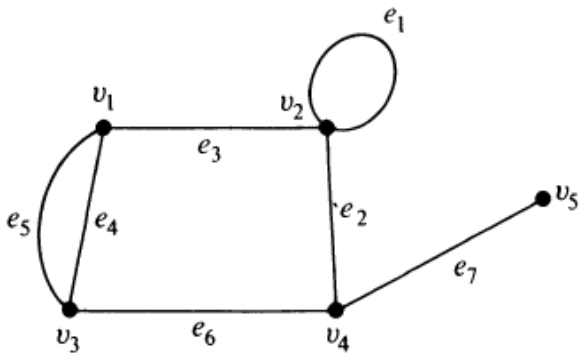


Fig. 1-1 Graph with five vertices and seven edges.

the definition allows more than one edge associated with a given pair of vertices, for example, edges e_4 and e_5 in Fig. 1-1. Such edges are referred to as *parallel edges*.

A graph that has neither self-loops nor parallel edges is called a *simple graph*. In some graph-theory literature, a graph is defined to be only a simple graph, but in most engineering applications it is necessary that parallel edges and self-loops be allowed; this is why our definition includes graphs with self-loops and/or parallel edges. Some authors use the term *general graph* to emphasize that parallel edges and self-loops are allowed.

It should also be noted that, in drawing a graph, it is immaterial whether the lines are drawn straight or curved, long or short: what is important is the incidence between the edges and vertices. For example, the two graphs drawn

incidence between the edges and vertices. For example, the two graphs drawn in Figs. 1-2(a) and (b) are the same, because incidence between edges and vertices is the same in both cases.

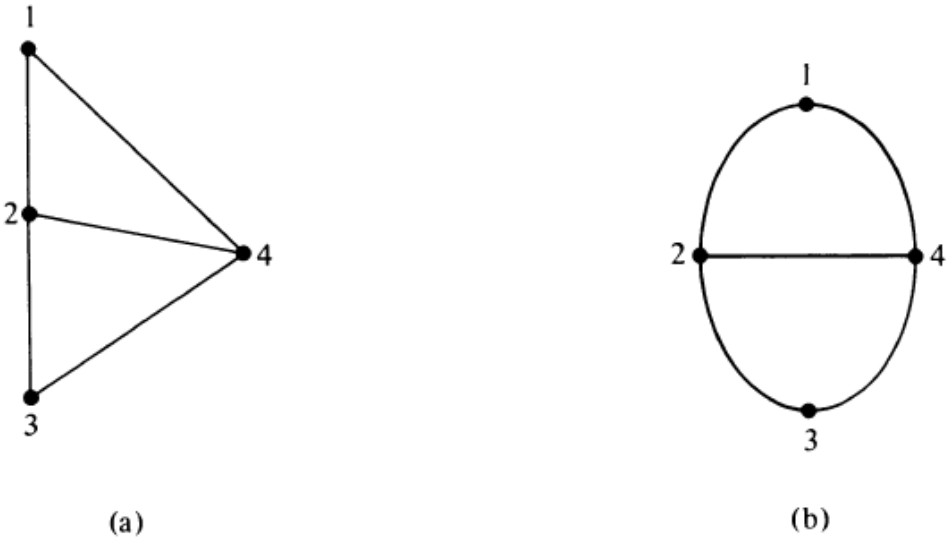


Fig. 1-2 Same graph drawn differently.

In a diagram of a graph, sometimes two edges may seem to intersect at a point that does not represent a vertex, for example, edges e and f in Fig. 1-3. Such edges should be thought of as being in different planes and thus having no common point. (Some authors break one of the two edges at such a crossing to emphasize this fact.)

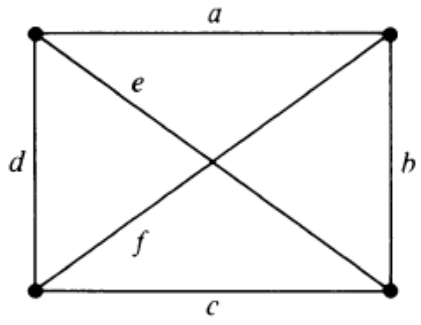
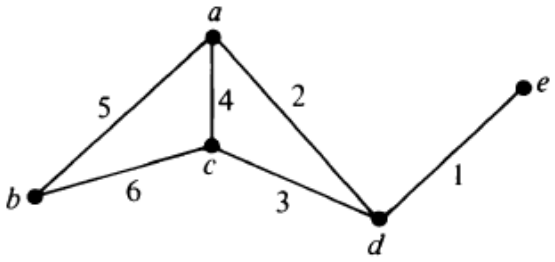


Fig. 1-3 Edges e and f have no common point.

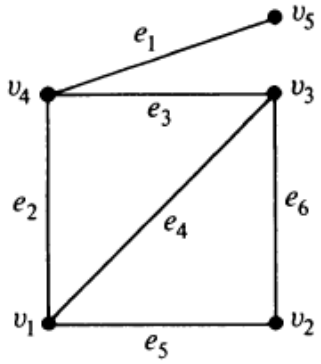
A graph is also called a *linear complex*, a *1-complex*, or a *one-dimensional complex*. A vertex is also referred to as a *node*, a *junction*, a *point*, *0-cell*, or an *0-simplex*. Other terms used for an edge are a *branch*, a *line*, an *element*, a *1-cell*, an *arc*, and a *1-simplex*. In this book we shall generally use the terms graph, vertex, and edge.

ISOMORPHISM

In geometry two figures are thought of as equivalent (and called congruent) if they have identical behavior in terms of geometric properties. Likewise, two graphs are thought of as equivalent (and called *isomorphic*) if they have identical behavior in terms of graph-theoretic properties. More precisely: Two graphs G and G' are said to be isomorphic (to each other) if there is a one-to-one correspondence between their vertices and between their edges such that the incidence relationship is preserved. In other words, suppose that edge e is incident on vertices v_1 and v_2 in G; then the corresponding edge e' in G' must be incident on the vertices v'_1 and v'_2 that correspond to



(a)



(b)

Fig. 2-1 Isomorphic graphs.

v_1 and v_2, respectively. For example, one can verify that the two graphs in Fig. 2-1 are isomorphic. The correspondence between the two graphs is as follows: The vertices a, b, c, d, and e correspond to v_1, v_2, v_3, v_4, and v_5, respectively. The edges 1, 2, 3, 4, 5, and 6 correspond to e_1, e_2, e_3, e_4, e_5, and e_6, respectively.

Except for the labels (i.e., names) of their vertices and edges, isomorphic graphs are the same graph, perhaps drawn differently. As indicated in Chapter 1, a given graph can be drawn in many different ways. For example, Fig. 2-2 shows two different ways of drawing the same graph.

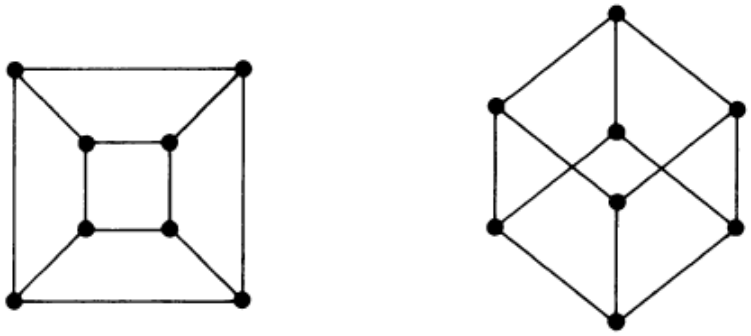
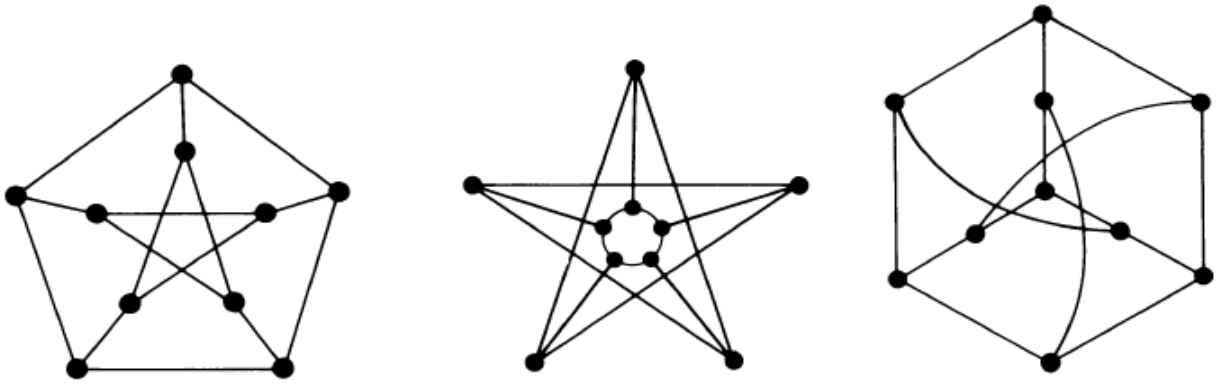


Fig. 2-2 Isomorphic graphs.

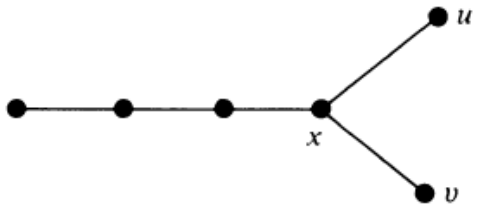
It is not always an easy task to determine whether or not two given graphs are isomorphic. For instance, the three graphs shown in Fig. 2-3 are all isomorphic, but just by looking at them you cannot tell that. It is left as an exercise for the reader to show, by redrawing and labeling the vertices and edges, that the three graphs in Fig. 2-3 are isomorphic (see Problem 2-3).

It is immediately apparent by the definition of isomorphism that two isomorphic graphs must have

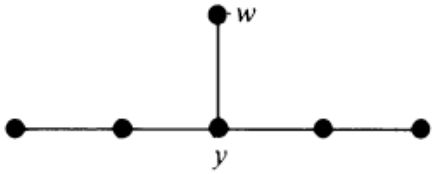
1. The same number of vertices.
2. The same number of edges.
3. An equal number of vertices with a given degree.



(a) (b) (c)



(a)



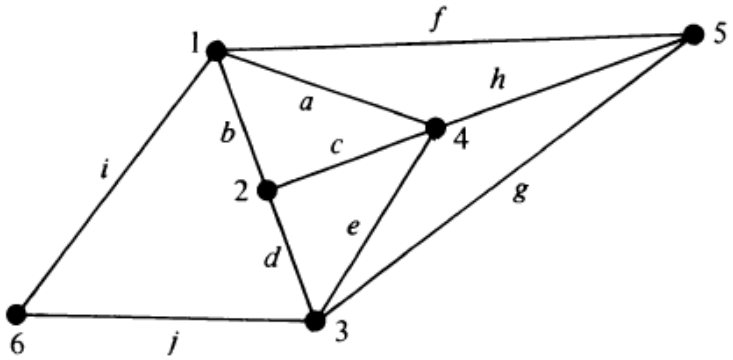
(b)

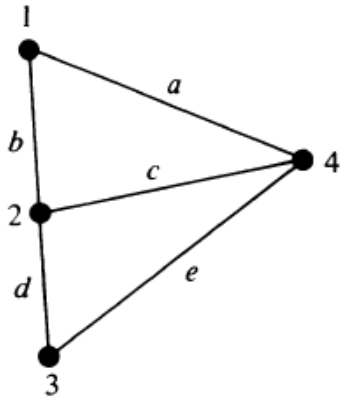
However, these conditions are by no means sufficient. For instance, the two graphs shown in Fig. 2-4 satisfy all three conditions, yet they are not isomorphic. That the graphs in Figs. 2-4(a) and (b) are not isomorphic can be shown as follows: If the graph in Fig. 2-4(a) were to be isomorphic to the one in (b), vertex x must correspond to y, because there are no other vertices of degree three. Now in (b) there is only one pendant vertex, w, adjacent to y, while in (a) there are two pendant vertices, u and v, adjacent to x.

Finding a simple and efficient criterion for detection of isomorphism is still actively pursued and is an important unsolved problem in graph theory. In Chapter 11 we shall discuss various proposed algorithms and their programs for automatic detection of isomorphism by means of a computer. For now, we move to a different topic.

SUBGRAPHS

A graph g is said to be a *subgraph* of a graph G if all the vertices and all the edges of g are in G, and each edge of g has the same end vertices in g as in G. For instance, the graph in Fig. 2-5(b) is a subgraph of the one in Fig. 2-5(a). (Obviously, when considering a subgraph, the original graph must





not be altered by identifying two distinct vertices, or by adding new edges or vertices.) The concept of subgraph is akin to the concept of subset in set theory. A subgraph can be thought of as being contained in (or a part of) another graph. The symbol from set theory, $g \subset G$, is used in stating "g is a subgraph of G."

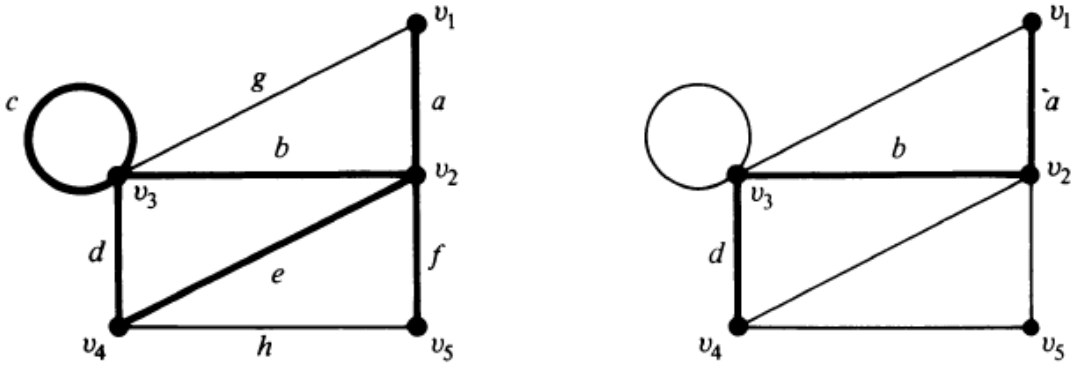
The following observations can be made immediately:

1. Every graph is its own subgraph.
2. A subgraph of a subgraph of G is a subgraph of G.
3. A single vertex in a graph G is a subgraph of G.
4. A single edge in G, together with its end vertices, is also a subgraph of G.

Edge-Disjoint Subgraphs: Two (or more) subgraphs g_1 and g_2 of a graph G are said to be *edge disjoint* if g_1 and g_2 do not have any edges in common. For example, the two graphs in Figs. 2-7(a) and (b) are edge-disjoint subgraphs of the graph in Fig. 2-6. Note that although edge-disjoint graphs do not have any edge in common, they may have vertices in common. Subgraphs that do not even have vertices in common are said to be *vertex disjoint*. (Obviously, graphs that have no vertices in common cannot possibly have edges in common.)

WALKS, PATHS, AND CIRCUITS

A *walk* is defined as a finite alternating sequence of vertices and edges, beginning and ending with vertices, such that each edge is incident with the vertices preceding and following it. No edge appears (is covered or traversed) more than once in a walk. A vertex, however, may appear more than once. In Fig. 2-8(a), for instance, $v_1\ a\ v_2\ b\ v_3\ c\ v_3\ d\ v_4\ e\ v_2\ f\ v_5$ is a walk shown with



(a) An Open Walk (b) A Path of Length Three

Fig. 2-8 A walk and a path.

heavy lines. A walk is also referred to as an *edge train* or a *chain*. The set of vertices and edges constituting a given walk in a graph G is clearly a subgraph of G.

Vertices with which a walk begins and ends are called its *terminal vertices*. Vertices v_1 and v_5 are the terminal vertices of the walk shown in Fig. 2-8(a). It is possible for a walk to begin and end at the same vertex. Such a walk is called a *closed walk*. A walk that is not closed (i.e., the terminal vertices are distinct) is called an *open walk* [Fig. 2-8(a)].

An open walk in which no vertex appears more than once is called a *path* (or a *simple path* or an *elementary path*). In Fig. 2-8, $v_1\ a\ v_2\ b\ v_3\ d\ v_4$ is a path, whereas $v_1\ a\ v_2\ b\ v_3\ c\ v_3\ d\ v_4\ e\ v_2\ f\ v_5$ is not a path. In other words, a path does not intersect itself. The number of edges in a path is called the *length of a path*. It immediately follows, then, that an edge which is not a self-loop is a path of length one. It should also be noted that a self-loop can be included in a walk but not in a path (Fig. 2-8).

The terminal vertices of a path are of degree one, and the rest of the vertices (called *intermediate vertices*) are of degree two. This degree, of course, is counted only with respect to the edges included in the path and not the entire graph in which the path may be contained.

A closed walk in which no vertex (except the initial and the final vertex) appears more than once is called a *circuit*. That is, a circuit is a closed, non-

Fig. 2-9 Three different circuits.

intersecting walk. In Fig. 2-8(a), $v_2\ b\ v_3\ d\ v_4\ e\ v_2$ is, for example, a circuit. Three different circuits are shown in Fig. 2-9. Clearly, every vertex in a circuit is of degree two; again, if the circuit is a subgraph of another graph, one must count degrees contributed by the edges in the circuit only.

A circuit is also called a *cycle*, *elementary cycle*, *circular path*, and *polygon*. In electrical engineering a circuit is sometimes referred to as a *loop*—not to be confused with self-loop. (Every self-loop is a circuit, but not every circuit is a self-loop.)

The definitions in this section are summarized in Fig. 2-10. The arrows are in the direction of increasing restriction.

You may have observed that although the concepts of a path and a circuit are very simple, the formal definition becomes involved.

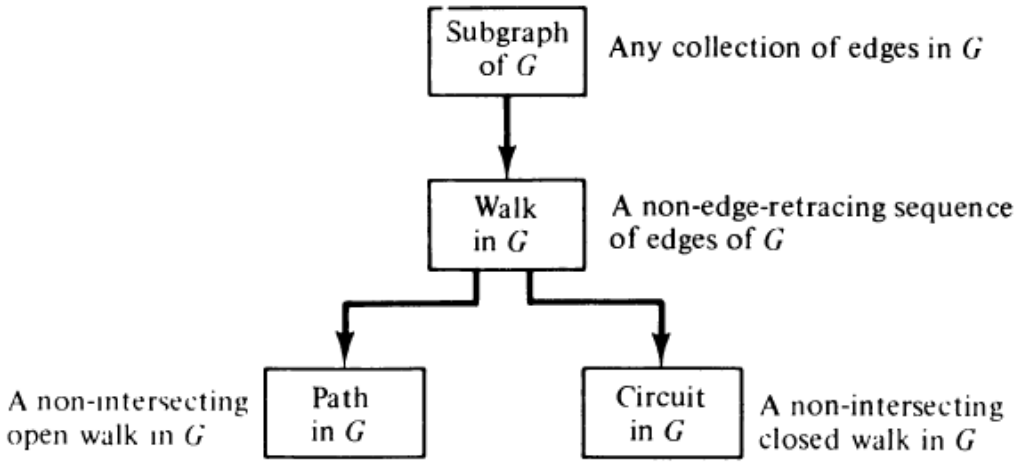


Fig. 2-10 Walks, paths, and circuits as subgraphs.

CONNECTED GRAPHS, DISCONNECTED GRAPHS, AND COMPONENTS

Intuitively, the concept of *connectedness* is obvious. A graph is connected if we can reach any vertex from any other vertex by traveling along the edges. More formally:

A graph G is said to be *connected* if there is at least one path between every pair of vertices in G. Otherwise, G is *disconnected*. For instance, the graph in Fig. 2-8(a) is connected, but the one in Fig. 2-11 is disconnected. A null graph of more than one vertex is disconnected (Fig. 1-12).

It is easy to see that a disconnected graph consists of two or more connected graphs. Each of these connected subgraphs is called a *component*. The graph in Fig. 2-11 consists of two components. Another way of looking at a component is as follows: Consider a vertex v_i in a disconnected graph G. By definition, not all vertices of G are joined by paths to v_i. Vertex v_i and all the vertices of G that have paths to v_i, together with all the edges incident on them, form a component. Obviously, a component itself is a graph.

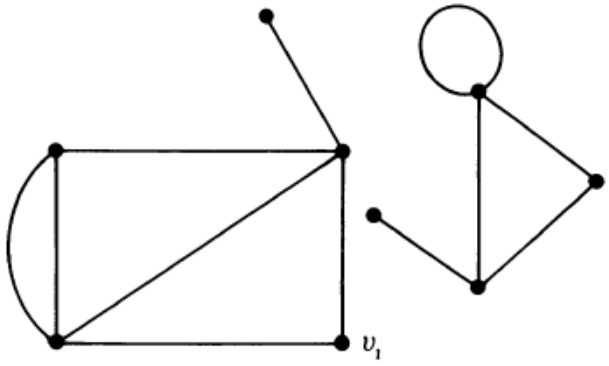


Fig. 2-11 A disconnected graph with two components.

THEOREM 2-1

A graph G is disconnected if and only if its vertex set V can be partitioned into two nonempty, disjoint subsets V_1 and V_2 such that there exists no edge in G whose one end vertex is in subset V_1 and the other in subset V_2.

Proof: Suppose that such a partitioning exists. Consider two arbitrary vertices a and b of G, such that $a \in V_1$ and $b \in V_2$. No path can exist between vertices a and b; otherwise, there would be at least one edge whose one end vertex would be in V_1 and the other in V_2. Hence, if a partition exists, G is not connected.

Conversely, let G be a disconnected graph. Consider a vertex a in G. Let V_1 be the set of all vertices that are joined by paths to a. Since G is disconnected, V_1 does not include all vertices of G. The remaining vertices will form a (nonempty) set V_2. No vertex in V_1 is joined to any in V_2 by an edge. Hence the partition. ■

Two interesting and useful results involving connectedness are:

THEOREM 2-2

If a graph (connected or disconnected) has exactly two vertices of odd degree, there must be a path joining these two vertices.

Proof: Let G be a graph with all even vertices† except vertices v_1 and v_2, which are odd. From Theorem 1-1, which holds for every graph and therefore for every component of a disconnected graph, no graph can have an odd number of odd vertices. Therefore, in graph G, v_1 and v_2 must belong to the same component, and hence must have a path between them. ■

THEOREM 2-3

A simple graph (i.e., a graph without parallel edges or self-loops) with n vertices and k components can have at most $(n - k)(n - k + 1)/2$ edges.

Proof: Let the number of vertices in each of the k components of a graph G be $n_1, n_2, \ldots, n_k$. Thus we have

$$n_1 + n_2 + \cdots + n_k = n,$$
$$n_i \geq 1.$$

†For brevity, a vertex with odd degree is called an *odd vertex*, and a vertex with even degree an *even vertex*.

The proof of the theorem depends on an algebraic inequality†

$$\sum_{i=1}^{k} n_i^2 \leq n^2 - (k - 1)(2n - k). \tag{2-1}$$

Now the maximum number of edges in the ith component of G (which is a simple connected graph) is $\frac{1}{2}n_i(n_i - 1)$. (See Problem 1-12.) Therefore, the maximum number of edges in G is

$$\frac{1}{2}\sum_{i=1}^{k}(n_i - 1)n_i = \frac{1}{2}\left(\sum_{i=1}^{k} n_i^2\right) - \frac{n}{2} \tag{2-2}$$

$$\leq \frac{1}{2}[n^2 - (k-1)(2n-k)] - \frac{n}{2}, \qquad \text{from (2-1)}$$

$$= \frac{1}{2}\cdot(n-k)(n-k+1). \quad \blacksquare \qquad (2\text{-}3)$$

It may be noted that Theorem 2-3 is a generalization of the result in Problem 1-12. The solution to Problem 1-12 is given by (2-3), where $k = 1$.

EULER GRAPHS

As mentioned in Chapter 1, graph theory was born in 1736 with Euler's famous paper in which he solved the Königsberg bridge problem. In the same paper, Euler posed (and then solved) a more general problem: In what type of graph G is it possible to find a closed walk running through every edge of G exactly once? Such a walk is now called an *Euler line*, and a graph that consists of an Euler line is called an *Euler graph*. More formally:

If some closed walk in a graph contains all the edges of the graph, then the walk is called an *Euler line* and the graph an *Euler graph*.

By its very definition a walk is always connected. Since the Euler line (which is a walk) contains all the edges of the graph, an *Euler graph* is always connected, except for any isolated vertices the graph may have. Since isolated vertices do not contribute anything to the understanding of an Euler graph, it is hereafter assumed that Euler graphs do not have any isolated vertices and are therefore connected.

Now we shall state and prove an important theorem, which will enable us to tell immediately whether or not a given graph is an Euler graph.

†*Proof:* $\sum_{i=1}^{k}(n_i - 1) = n - k$. Squaring both sides,

$$\left(\sum_{i-1}^{k}(n_i - 1)\right)^2 = n^2 + k^2 - 2nk$$

or $\sum_{i=1}^{k}(n_i^2 - 2n_i) + k +$ nonnegative cross terms $= n^2 + k^2 - 2nk$ because $(n_i - 1) \geq 0$, for all i. Therefore, $\sum_{i=1}^{k} n_i^2 \leq n^2 + k^2 - 2nk - k + 2n = n^2 - (k-1)(2n-k)$. ■

THEOREM 2-4

A given connected graph G is an Euler graph if and only if all vertices of G are of even degree.

Proof: Suppose that G is an Euler graph. It therefore contains an Euler line (which is a closed walk). In tracing this walk we observe that every time the walk meets a vertex v it goes through two "new" edges incident on v—with one we "entered" v and with the other "exited." This is true not only of all intermediate vertices of the walk but also of the terminal vertex, because we "exited" and "entered" the same vertex at the beginning and end of the walk, respectively. Thus if G is an Euler graph, the degree of every vertex is even.

To prove the sufficiency of the condition, assume that all vertices of G are of even degree. Now we construct a walk starting at an arbitrary vertex v and going through the edges of G such that no edge is traced more than once. We continue tracing as far as possible. Since every vertex is of even degree, we can exit from every vertex we enter; the tracing cannot stop at any vertex but v. And since v is also of even degree, we shall eventually reach v when the tracing comes to an end. If this closed walk h we just traced includes all the edges of G, G is an Euler graph. If not, we remove from G all the edges in h and obtain a subgraph h' of G formed by the remaining edges. Since both G and h have all their vertices of even degree, the degrees of the vertices of h' are also even. Moreover, h' must touch h at least at one vertex a, because G is connected. Starting from a, we can again construct a new walk in graph h'. Since all the vertices of h' are of even degree, this walk in h' must terminate at vertex a; but this walk in h' can be combined with h to form a new walk, which starts and ends at vertex v and has more edges than h. This process can be repeated until we obtain a closed walk that traverses all the edges of G. Thus G is an Euler graph. ■

Königsberg Bridge Problem: Now looking at the graph of the Königsberg bridges (Fig. 1-5), we find that not all its vertices are of even degree. Hence, it is not an Euler graph. Thus it is not possible to walk over each of the seven bridges exactly once and return to the starting point.

One often encounters Euler lines in various puzzles. The problem common to these puzzles is to find how a given picture can be drawn in one continuous line without retracing and without lifting the pencil from the paper. Two such pictures are shown in Fig. 2-12. The drawing in Fig. 2-12(a) is called *Mohammed's scimitars* and is believed to have come from the Arabs. The one in Fig. 2-12(b) is, of course, the *star of David.* (Equal time!)

In defining an Euler line some authors drop the requirement that the walk be closed. For example, the walk a 1 c 2 d 3 a 4 b 5 d 6 e 7 b in Fig. 2-13, which includes all the edges of the graph and does not retrace any edge, is not closed. The initial vertex is a and the final vertex is b. We shall call such an open walk that includes (or traces or covers) all edges of a graph without retracing any edge a *unicursal line* or an *open Euler line*. A (connected) graph that has a unicursal line will be called a *unicursal graph*.

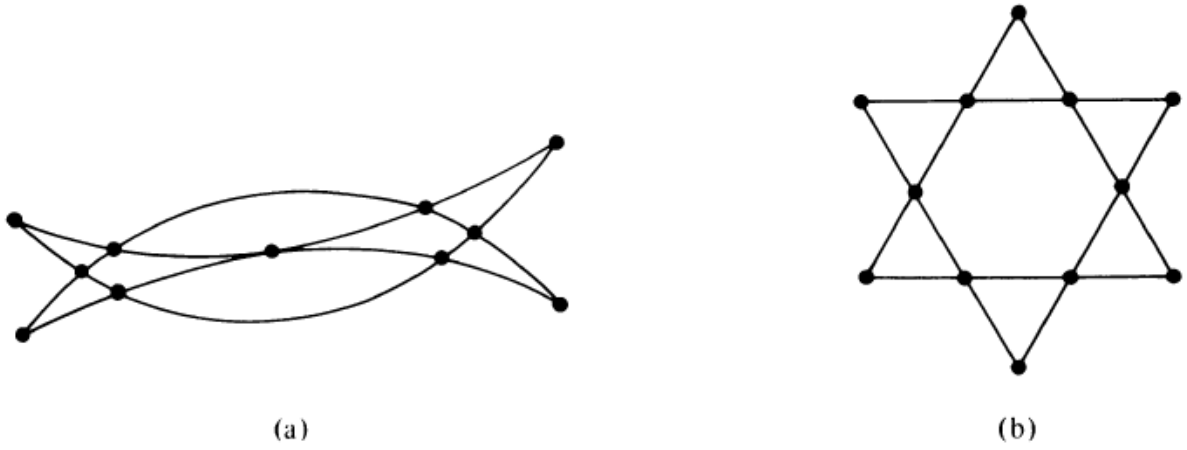


Fig. 2-12 Two Euler graphs.

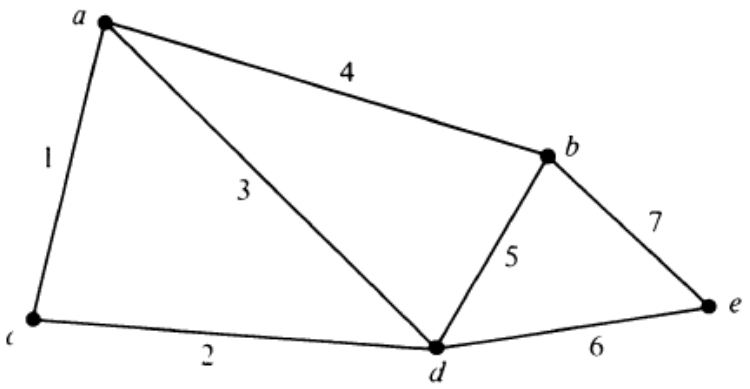


Fig. 2-13 Unicursal graph.

It is clear that by adding an edge between the initial and final vertices of a unicursal line we shall get an Euler line. Thus a connected graph is unicursal if and only if it has exactly two vertices of odd degree. This observation can be generalized as follows:

THEOREM 2-5

In a connected graph G with exactly $2k$ odd vertices, there exist k edge-disjoint subgraphs such that they together contain all edges of G and that each is a unicursal graph.

Proof: Let the odd vertices of the given graph G be named $v_1, v_2, \ldots, v_k$; $w_1, w_2, \ldots, w_k$ in any arbitrary order. Add k edges to G between the vertex pairs $(v_1, w_1), (v_2, w_2), \ldots, (v_k, w_k)$ to form a new graph G'.

Since every vertex of G' is of even degree, G' consists of an Euler line ρ. Now if we remove from ρ the k edges we just added (no two of these edges are incident on the same vertex), ρ will be split into k walks, each of which is a unicursal line: The first removal will leave a single unicursal line; the second removal will split that into two unicursal lines; and each successive removal will split a unicursal line into two unicursal lines, until there are k of them. Thus the theorem. ■

THEOREM 2-6

A connected graph G is an Euler graph if and only if it can be decomposed into circuits.

Proof: Suppose graph G can be decomposed into circuits; that is, G is a union of edge-disjoint circuits. Since the degree of every vertex in a circuit is two, the degree of every vertex in G is even. Hence G is an Euler graph.

Conversely, let G be an Euler graph. Consider a vertex v_1. There are at least two edges incident at v_1. Let one of these edges be between v_1 and v_2. Since vertex v_2 is also of even degree, it must have at least another edge, say between v_2 and v_3. Proceeding in this fashion, we eventually arrive at a vertex that has previously been traversed, thus forming a circuit Γ. Let us remove Γ from G. All vertices in the remaining graph (not necessarily connected) must also be of even degree. From the remaining graph remove another circuit in exactly the same way as we removed Γ from G. Continue this process until no edges are left. Hence the theorem. ■

Arbitrarily Traceable Graphs: Consider the graph in Fig. 2-17, which is an Euler graph. Suppose that we start from vertex *a* and trace the path *a b c*.

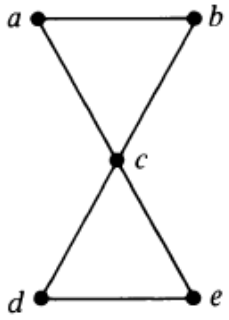


Fig. 2-17 Arbitrarily traceable graph from c.

Now at c we have the choice of going to a, d, or e. If we took the first choice, we would only trace the circuit $a\ b\ c\ a$, which is not an Euler line. Thus, starting from a, we cannot trace the entire Euler line simply by moving along any edge that has not already been traversed. This raises the following interesting question:

HAMILTONIAN PATHS AND CIRCUITS

An Euler line of a connected graph was characterized by the property of being a closed walk that traverses *every edge* of the graph (exactly once). A *Hamiltonian circuit* in a connected graph is defined as a closed walk that traverses *every vertex* of G exactly once, except of course the starting vertex, at which the walk also terminates. For example, in the graph of Fig. 2-20(a)

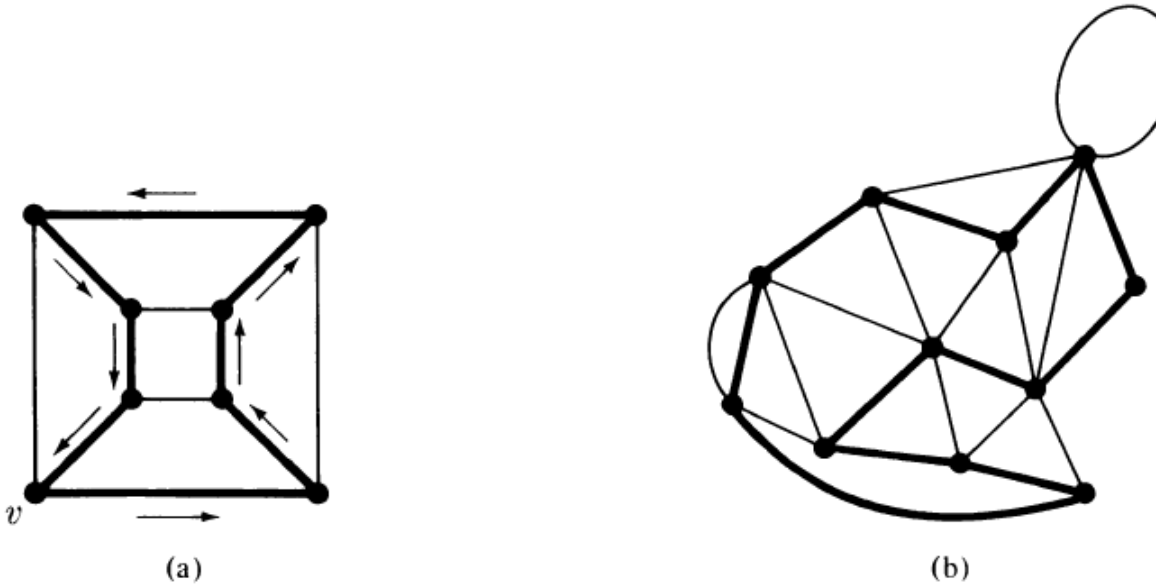


Fig. 2-20 Hamiltonian circuits.

starting at vertex v, if one traverses along the edges shown in heavy lines—passing through each vertex exactly once—one gets a Hamiltonian circuit. A Hamiltonian circuit for the graph in Fig. 2-20(b) is also shown by heavy lines. More formally:

A circuit in a connected graph G is said to be Hamiltonian if it includes every vertex of G. Hence a Hamiltonian circuit in a graph of n vertices consists of exactly n edges.

Obviously, not every connected graph has a Hamiltonian circuit. For example, neither of the graphs shown in Figs. 2-17 and 2-18 has a Hamiltonian circuit. This raises the question: What is a necessary and sufficient condition for a connected graph G to have a Hamiltonian circuit?

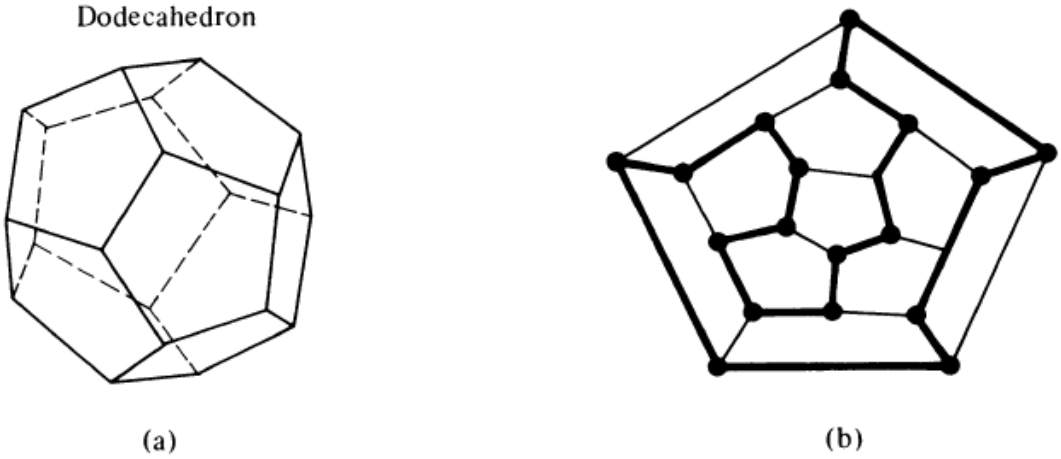


Fig. 2-21 Dodecahedron and its graph shown with a Hamiltonian circuit.

This problem, first posed by the famous Irish mathematician Sir William Rowan Hamilton in 1859, is still unsolved. As was mentioned in Chapter 1, Hamilton made a regular dodecahedron of wood [see Fig. 2-21(a)], each of whose 20 corners was marked with the name of a city. The puzzle was to start from any city and find a route along the edge of the dodecahedron that passes through every city exactly once and returns to the city of origin. The graph of the dodecahedron is given in Fig. 2-21(b), and one of many such routes (a Hamiltonian circuit) is shown by heavy lines.

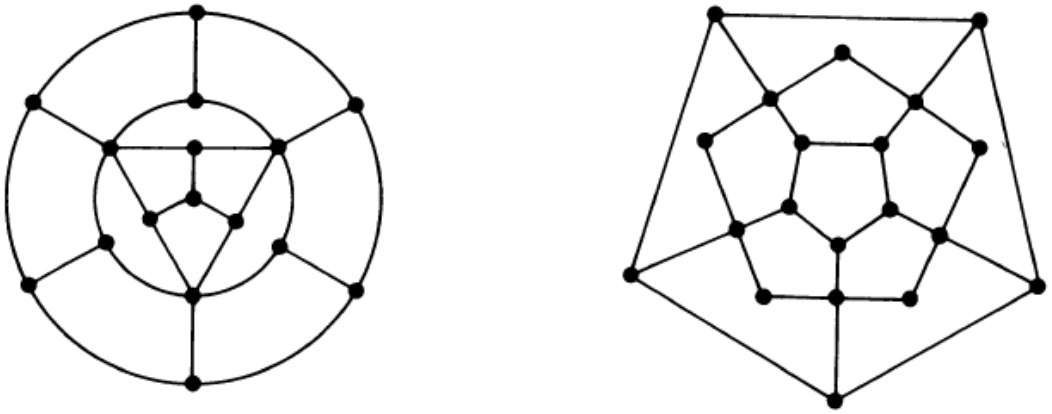
The resemblance between the problem of an Euler line and that of a Hamiltonian circuit is deceptive. The latter is infinitely more complex. Although one can find Hamiltonian circuits in many specific graphs, such as those shown in Figs. 2-20 and 2-21, there is no known criterion we can apply

to determine the existence of a Hamiltonian circuit in general. There are, however, certain types of graphs that always contain Hamiltonian circuits, as will be presently shown.

Hamiltonian Path: If we remove any one edge from a Hamiltonian circuit, we are left with a path. This path is called a *Hamiltonian path*. Clearly, a Hamiltonian path in a graph G traverses every vertex of G. Since a Hamiltonian path is a subgraph of a Hamiltonian circuit (which in turn is a subgraph of another graph), every graph that has a Hamiltonian circuit also has a Hamiltonian path. There are, however, many graphs with Hamiltonian paths that have no Hamiltonian circuits (Problem 2-23). The length of a Hamiltonian path (if it exists) in a connected graph of n vertices is $n - 1$.

In considering the existence of a Hamiltonian circuit (or path), we need only consider simple graphs. This is because a Hamiltonian circuit (or path) traverses every vertex exactly once. Hence it cannot include a self-loop or a set of parallel edges. Thus a general graph may be made simple by removing parallel edges and self-loops before looking for a Hamiltonian circuit in it.

It is left as an exercise for the reader to show that neither of the two graphs



(a) (b)

Fig. 2-22 Graphs without Hamiltonian circuits.

shown in Fig. 2-22 has a Hamiltonian circuit (or Hamiltonian path). See Problem 2-24.

shown in Fig. 2-22 has a Hamiltonian circuit (or Hamiltonian path). See Problem 2-24.

What general class of graphs is guaranteed to have a Hamiltonian circuit? Complete graphs of three or more vertices constitute one such class.

Complete Graph: A simple graph in which there exists an edge between every pair of vertices is called a *complete graph*. Complete graphs of two, three, four, and five vertices are shown in Fig. 2-23. A complete graph is

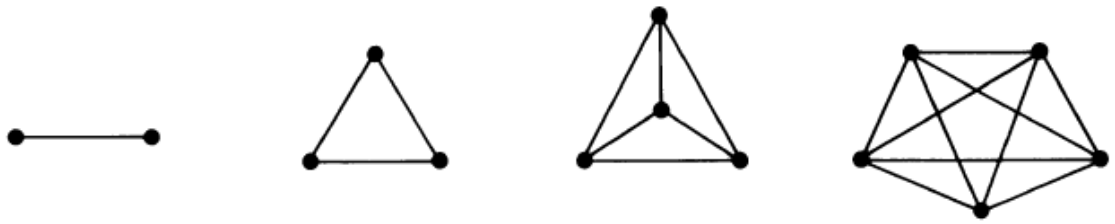


Fig. 2-23 Complete graphs of two, three, four, and five vertices.

sometimes also referred to as a *universal graph* or a *clique*. Since every vertex is joined with every other vertex through one edge, the degree of every vertex is $n - 1$ in a complete graph G of n vertices. Also the total number of edges in G is $n(n - 1)/2$. See Problem 1-12.

It is easy to construct a Hamiltonian circuit in a complete graph of n vertices. Let the vertices be numbered $v_1, v_2, \ldots, v_n$. Since an edge exists between any two vertices, we can start from v_1 and traverse to v_2, and v_3, and so on to v_n, and finally from v_n to v_1. This is a Hamiltonian circuit.

Number of Hamiltonian Circuits in a Graph: A given graph may contain more than one Hamiltonian circuit. Of interest are all the edge-disjoint Hamiltonian circuits in a graph. The determination of the exact number of *edge-disjoint* Hamiltonian circuits (or paths) in a graph in general is also an unsolved problem. However, the number of edge-disjoint Hamiltonian circuits in a complete graph with odd number of vertices is given by Theorem 2-8.

THEOREM 2-8

In a complete graph with n vertices there are $(n - 1)/2$ edge-disjoint Hamiltonian circuits, if n is an odd number ≥ 3.

Proof: A complete graph G of n vertices has $n(n-1)/2$ edges, and a Hamiltonian circuit in G consists of n edges. Therefore, the number of edge-disjoint Hamiltonian circuits in G cannot exceed $(n-1)/2$. That there are $(n-1)/2$ edge-disjoint Hamiltonian circuits, when n is odd, can be shown as follows:

The subgraph (of a complete graph of n vertices) in Fig. 2-24 is a Hamiltonian circuit. Keeping the vertices fixed on a circle, rotate the polygonal pattern clockwise

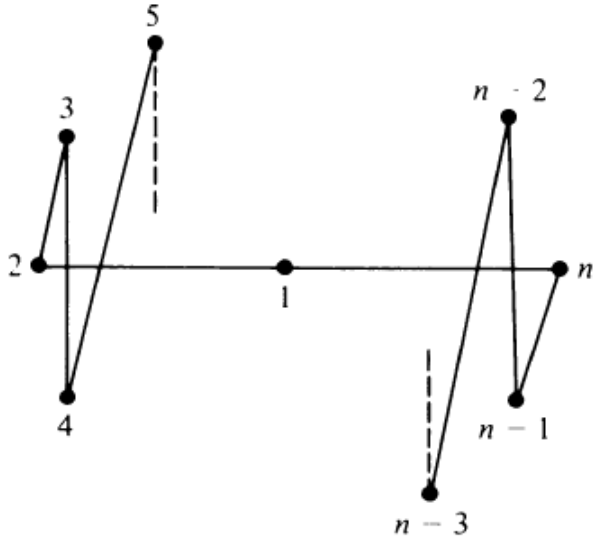


Fig. 2-24 Hamiltonian circuit; n is odd.

by $360/(n-1), 2\cdot 360/(n-1), 3\cdot 360/(n-1), \ldots, (n-3)/2\cdot 360/(n-1)$ degrees. Observe that each rotation produces a Hamiltonian circuit that has no edge in common with any of the previous ones. Thus we have $(n-3)/2$ new Hamiltonian circuits, all edge disjoint from the one in Fig. 2-24 and also edge disjoint among themselves. Hence the theorem. ■

This theorem enables us to solve the problem of the seating arrangement at a round table, introduced in Chapter 1, as follows:

Representing a member x by a vertex and the possibility of his sitting next to another member y by an edge between x and y, we construct a graph G. Since every member is allowed to sit next to any other member, G is a complete graph of nine vertices—nine being the number of people to be seated around the table. Every seating arrangement around the table is clearly a Hamiltonian circuit.

The first day of their meeting they can sit in any order, and it will be a Hamiltonian circuit H_1. The second day, if they are to sit such that every member must have different neighbors, we have to find another Hamiltonian circuit H_2 in G, with an entirely different set of edges from those in H_1; that is,

H_1 and H_2 are edge-disjoint Hamiltonian circuits. From Theorem 2-8 the number of edge-disjoint Hamiltonian circuits in G is four; therefore, only four such arrangements exist among nine people.

Another interesting result on the question of existence of Hamiltonian circuits in a graph, obtained by G. A. Dirac, is:

THEOREM 2-9

A sufficient (but by no means necessary) condition for a simple graph G to have a Hamiltonian circuit is that the degree of every vertex in G be at least $n/2$, where n is the number of vertices in G.

Proof: For proof the reader is referred to the original paper by Dirac [2-3].

TREES

A *tree* is a connected graph without any circuits. The graph in Fig. 3-1, for instance, is a tree. Trees with one, two, three, and four vertices are shown in Fig. 3-2. As pointed out in Chapter 1, a graph must have at least one vertex, and therefore so must a tree. Some authors allow the *null tree*, a tree without any vertices. We have excluded such an entity from being a tree. Similarly, as we are considering only finite graphs, our trees are also finite.

It follows immediately from the definition that a tree has to be a simple graph, that is, having neither a self-loop nor parallel edges (because they both form circuits).

Trees appear in numerous instances. The genealogy of a family is often

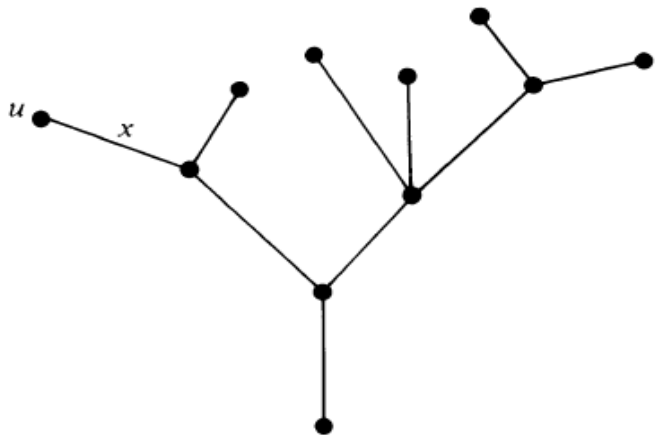


Fig. 3-1 Tree.

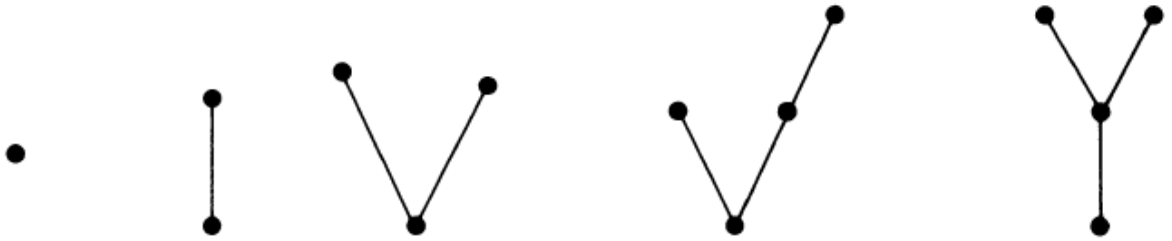


Fig. 3-2 Trees with one, two, three, and four vertices.

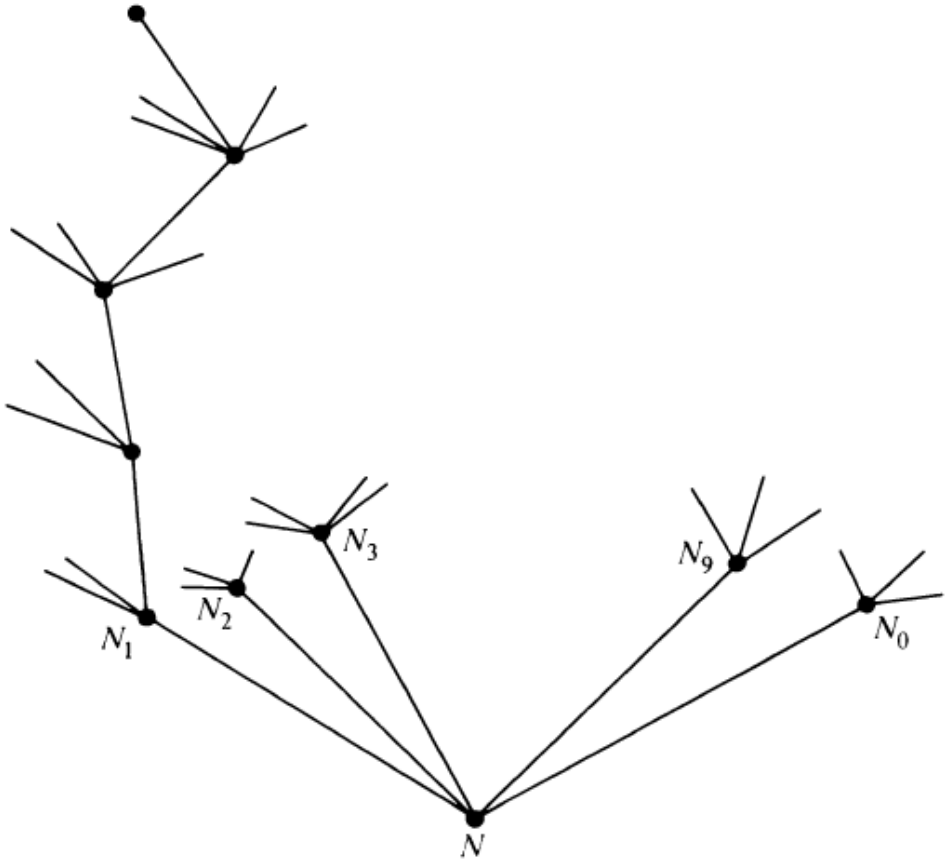


Fig. 3-3 Decision tree.

represented by means of a tree (in fact the term *tree* comes from *family tree*). A river with its tributaries and subtributaries can be represented by a tree. The sorting of mail according to zip code and the sorting of punched cards are done according to a tree (called *decision tree* or *sorting tree*).

Figure 3-3 might represent the flow of mail. All the mail arrives at some local office, vertex N. The most significant digit in the zip code is read at N, and the mail is divided into 10 piles $N_1, N_2, \ldots, N_9$, and N_0, depending on

the most significant digit. Each pile is further divided into 10 piles according to the second most significant digit, and so on, till the mail is subdivided into 10^5 possible piles, each representing a unique five-digit zip code.

In many sorting problems we have only two alternatives (instead of 10 as in the preceding example) at each intermediate vertex, representing a dichotomy, such as large or small, good or bad, 0 or 1. Such a decision tree with two choices at each vertex occurs frequently in computer programming and switching theory. We shall deal with such trees and their applications in Section 3-5. Let us first obtain a few simple but important theorems on the general properties of trees.

SOME PROPERTIES OF TREES

THEOREM 3-1

There is one and only one path between every pair of vertices in a tree, T.

Proof: Since T is a connected graph, there must exist at least one path between every pair of vertices in T. Now suppose that between two vertices a and b of T there are two distinct paths. The union of these two paths will contain a circuit and T cannot be a tree. ■

Conversely:

THEOREM 3-2

If in a graph G there is one and only one path between every pair of vertices, G is a tree.

Proof: Existence of a path between every pair of vertices assures that G is connected. A circuit in a graph (with two or more vertices) implies that there is at least one pair of vertices a, b such that there are two distinct paths between a and b. Since G has one and only one path between every pair of vertices, G can have no circuit. Therefore, G is a tree. ■

THEOREM 3-3

A tree with n vertices has $n - 1$ edges.

Proof: The theorem will be proved by induction on the number of vertices.

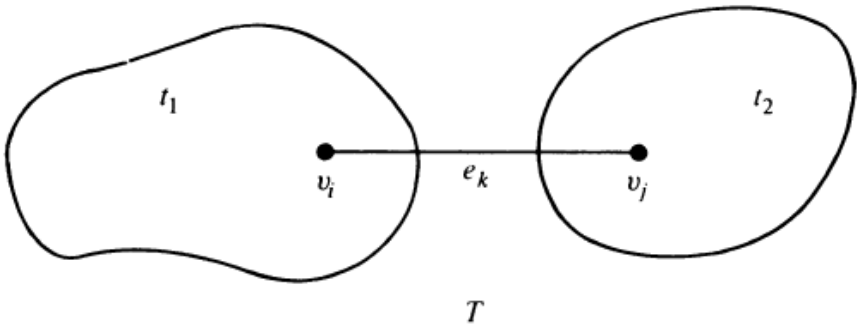


Fig. 3-4 Tree T with n vertices.

It is easy to see that the theorem is true for $n = 1$, 2, and 3 (see Fig. 3-2). Assume that the theorem holds for all trees with fewer than n vertices.

Let us now consider a tree T with n vertices. In T let e_k be an edge with end vertices v_i and v_j. According to Theorem 3-1, there is no other path between v_i and v_j except e_k. Therefore, deletion of e_k from T will disconnect the graph, as shown in Fig. 3-4. Furthermore, $T - e_k$ consists of exactly two components, and since there were no circuits in T to begin with, each of these components is a tree. Both these trees, t_1 and t_2, have fewer than n vertices each, and therefore, by the induction hypothesis, each contains one less edge than the number of vertices in it. Thus $T - e_k$ consists of $n - 2$ edges (and n vertices). Hence T has exactly $n - 1$ edges. ■

THEOREM 3-4

Any connected graph with n vertices and $n - 1$ edges is a tree.

Proof: The proof of the theorem is left to the reader as an exercise (Problem 3-5).

You may have noticed another important feature of a tree: its vertices are connected together with the minimum number of edges. A connected graph is said to be *minimally connected* if removal of any one edge from it disconnects the graph. A minimally connected graph cannot have a circuit; otherwise, we could remove one of the edges in the circuit and still leave the graph connected. Thus a minimally connected graph is a tree. Conversely, if a connected graph G is not minimally connected, there must exist an edge e_i in G such that $G - e_i$ is connected. Therefore, e_i is in some circuit, which implies that G is not a tree. Hence the following theorem:

THEOREM 3-5

A graph is a tree if and only if it is minimally connected.

The significance of Theorem 3-5 is obvious. Intuitively, one can see that to interconnect n distinct points, the minimum number of line segments needed is $n - 1$. It requires no background in electrical engineering to realize

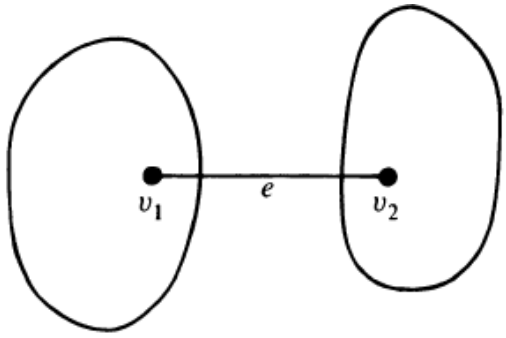


Fig. 3-5 Edge e added to $G = g_1 \cup g_2$.

that to short (electrically) n pins together, one needs at least $n - 1$ pieces of wire. The resulting structure, according to Theorem 3-5, is a tree.

We showed that a connected graph with n vertices and without any circuits has $n - 1$ edges. We can also show that a graph with n vertices which has no circuit and has $n - 1$ edges is always connected (i.e., it is a tree), in the following theorem.

THEOREM 3-6

A graph G with n vertices, $n - 1$ edges, and no circuits is connected.

Proof: Suppose there exists a circuitless graph G with n vertices and $n - 1$ edges which is disconnected. In that case G will consist of two or more circuitless components. Without loss of generality, let G consist of two components, g_1 and g_2. Add an edge e between a vertex v_1 in g_1 and v_2 in g_2 (Fig. 3-5). Since there was no path between v_1 and v_2 in G, adding e did not create a circuit. Thus $G \cup e$ is a circuitless, connected graph (i.e., a tree) of n vertices and n edges, which is not possible, because of Theorem 3-3. ■

The results of the preceding six theorems can be summarized by saying that the following are five different but equivalent definitions of a tree. That is, a graph G with n vertices is called a tree if

1. G is *connected* and is *circuitless*, or
2. G is *connected* and has $n - 1$ *edges*, or
3. G is *circuitless* and has $n - 1$ *edges*, or
4. There is *exactly one path* between every pair of vertices in G, or
5. G is a *minimally connected* graph.

PENDANT VERTICES IN A TREE

You must have observed that each of the trees shown in the figures has several pendant vertices (a pendant vertex was defined as a vertex of degree

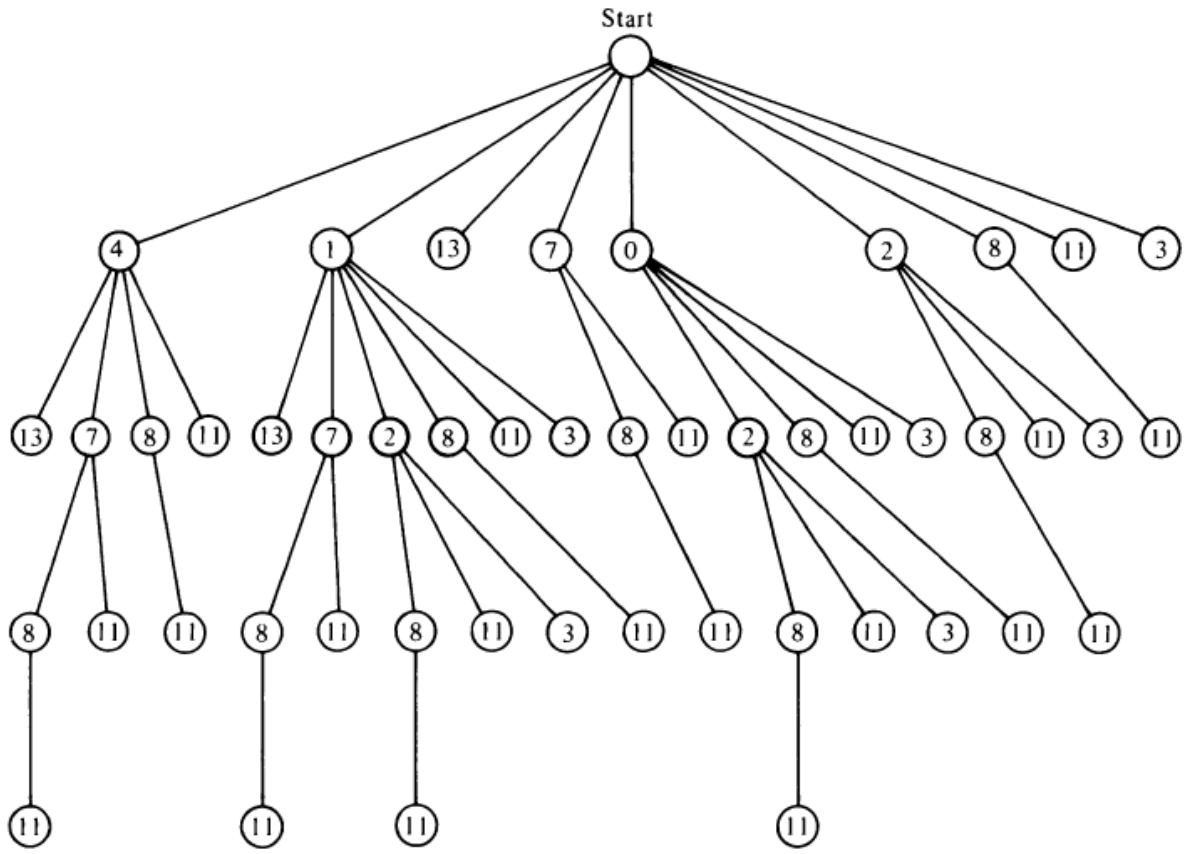


Fig. 3-6 Tree of the monotonically increasing sequences in 4, 1, 13, 7, 0, 2, 8, 11, 3.

one). The reason is that in a tree of n vertices we have $n - 1$ edges, and hence $2(n - 1)$ degrees to be divided among n vertices. Since no vertex can be of zero degree, we must have at least two vertices of degree one in a tree. This of course makes sense only if $n \geq 2$. More formally:

THEOREM 3-7

In any tree (with two or more vertices), there are at least two pendant vertices.

An Application: The following problem is used in teaching computer programming. Given a sequence of integers, no two of which are the same, find the largest monotonically increasing subsequence in it. Suppose that the sequence given to us is 4, 1, 13, 7, 0, 2, 8, 11, 3; it can be represented by a tree in which the vertices (except the start vertex) represent individual numbers in the sequence, and the path from the start vertex to a particular vertex v describes the monotonically increasing subsequence terminating in v. As shown in Fig. 3-6, this sequence contains four longest monotonically increasing subsequences, that is, (4, 7, 8, 11), (1, 7, 8, 11), (1, 2, 8, 11), and (0, 2, 8, 11). Each is of length four. Such a tree used in representing data is referred to as a data tree by computer programmers.

DISTANCE AND CENTERS IN A TREE

The tree in Fig. 3-7 has four vertices. Intuitively, it seems that vertex b is located more "centrally" than any of the other three vertices. We shall explore this idea further and see if in a tree there exists a "center" (or centers). Inherent in the concept of a center is the idea of "distance," so we must define distance before we can talk of a center.

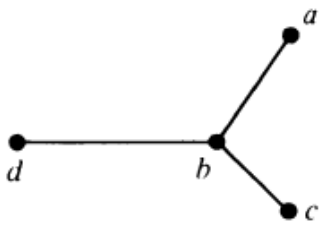


Fig. 3-7 Tree.

In a connected graph G, the *distance* $d(v_i, v_j)$ between two of its vertices v_i and v_j is the length of the shortest path (i.e., the number of edges in the shortest path) between them.

The definition of distance between any two vertices is valid for any connected graph (not necessarily a tree). In a graph that is not a tree, there are generally several paths between a pair of vertices. We have to enumerate all

these paths and find the length of the shortest one. (There may be several shortest paths.)

For instance, some of the paths between vertices v_1 and v_2 in Fig. 3-8 are (a, e), (a, c, f), (b, c, e), (b, f), (b, g, h), and (b, g, i, k). There are two shortest paths, (a, e) and (b, f), each of length two. Hence $d(v_1, v_2) = 2$.

In a tree, since there is exactly one path between any two vertices (Theorem 3-1), the determination of distance is much easier. For instance, in the tree of Fig. 3-7, $d(a, b) = 1$, $d(a, c) = 2$, $d(c, b) = 1$, and so on.

A Metric: Before we can legitimately call a function $f(x, y)$ of two variables a "distance" between them, this function must satisfy certain requirements. These are

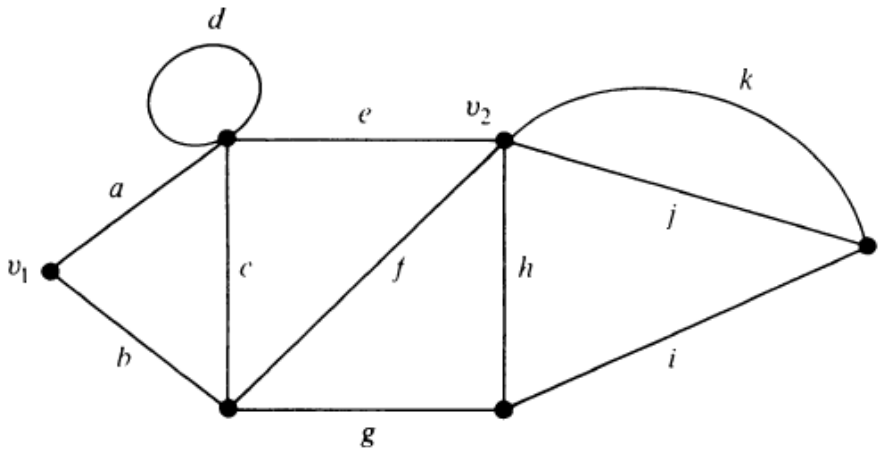


Fig. 3-8 Distance between v_1 and v_2 is two.

1. Nonnegativity: $f(x, y) \geq 0$, and $f(x, y) = 0$ if and only if $x = y$.
2. Symmetry: $f(x, y) = f(y, x)$.
3. Triangle inequality: $f(x, y) \leq f(x, z) + f(z, y)$ for any z.

A function that satisfies these three conditions is called a *metric*. That the distance $d(v_i, v_j)$ between two vertices of a connected graph satisfies conditions 1 and 2 is immediately evident. Since $d(v_i, v_j)$ is the length of the shortest path between vertices v_i and v_j, this path cannot be longer than another path between v_i and v_j, which goes through a specified vertex v_k. Hence $d(v_i, v_j) \leq d(v_i, v_k) + d(v_k, v_j)$. Therefore,

THEOREM 3-8

The distance between vertices of a connected graph is a metric.

Coming back to our original topic of relative location of different vertices in a tree, let us define another term called *eccentricity* (also referred to as *associated number* or *separation*) of a vertex in a graph.

The eccentricity $E(v)$ of a vertex v in a graph G is the distance from v to the vertex farthest from v in G; that is,

$$E(v) = \max_{v_i \in G} d(v, v_i).$$

A vertex with minimum eccentricity in graph G is called a *center* of G. The eccentricities of the four vertices in Fig. 3-7 are $E(a) = 2$, $E(b) = 1$, $E(c) = 2$, and $E(d) = 2$. Hence vertex b is the center of that tree. On the other hand, consider the tree in Fig. 3-9. The eccentricity of each of its six vertices is shown next to the vertex. This tree has two vertices having the same minimum eccentricity. Hence this tree has two centers. Some authors refer to such centers as *bicenters;* we shall call them just centers, because there will be no occasion for confusion.

The reader can easily verify that a graph, in general, has many centers. For example, in a graph that consists of just a circuit (a polygon), every vertex is a center. In the case of a tree, however, König [1-7] proved the following theorem:

THEOREM 3-9

Every tree has either one or two centers.

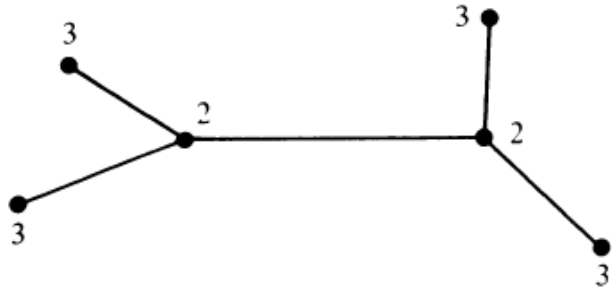


Fig 3-9 Eccentricities of the vertices of a tree.

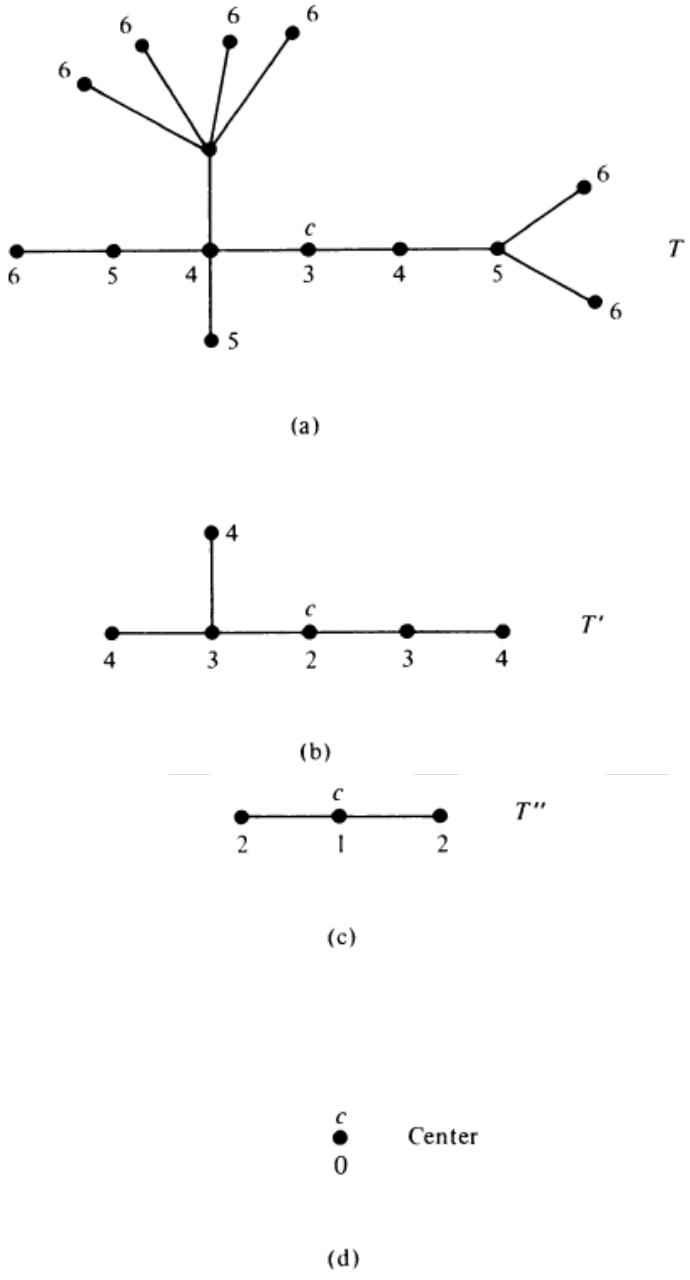


Fig. 3-10 Finding a center of a tree.

Proof: The maximum distance, max $d(v, v_i)$, from a given vertex v to any other vertex v_i occurs only when v_i is a pendant vertex. With this observation, let us start with a tree T having more than two vertices. Tree T must have two or more pendant vertices (Theorem 3-7). Delete all the pendant vertices from T. The resulting graph T' is still a tree. What about the eccentricities of the vertices in T'? A little deliberation will reveal that removal of all pendant vertices from T uniformly

reduced the eccentricities of the remaining vertices (i.e., vertices in T') by one. Therefore, all vertices that T had as centers will still remain centers in T'. From T' we can again remove all pendant vertices and get another tree T''. We continue this process (which is illustrated in Fig. 3-10) until there is left either a vertex (which is the center of T) or an edge (whose end vertices are the two centers of T). Thus the theorem. ■

COROLLARY

From the argument used in proving Theorem 3-9, we see that if a tree T has two centers, the two centers must be adjacent.

A Sociological Application: Suppose that the communication among a group of 14 persons in a society is represented by the graph in Fig. 3-10(a), where the vertices represent the persons and an edge represents the communication link between its two end vertices. Since the graph is connected, we know that all the members can be reached by any member, either directly or through some other members. But it is also important to note that the graph is a tree—minimally connected. The group cannot afford to lose any of the communication links.

The eccentricity of each vertex, $E(v)$, represents how close v is to the farthest member of the group. In Fig. 3-10(a), vertex c should be the leader of the group, if closeness of communication were the criterion for leadership.

Radius and Diameter: If a tree has a center (or two centers), does it have a radius also? Yes. The eccentricity of a center (which is the distance from the center of the tree to the farthest vertex) in a tree is defined as the *radius* of the tree. For instance, the radius of the tree in Fig. 3-10(a) is three. The *diameter* of a tree T, on the other hand, is defined as the length of the longest path in T. It is left as an exercise for the reader (Problem 3-6) to show that a radius in a tree is not necessarily half its diameter.

ROOTED AND BINARY TREES

A tree in which one vertex (called the *root*) is distinguished from all the others is called a *rooted tree*. For instance, in Fig. 3-3 vertex N, from where all the mail goes out, is distinguished from the rest of the vertices. Hence N can be considered the root of the tree, and so the tree is rooted. Similarly, in Fig. 3-6 the start vertex may be considered as the root of the tree shown. In a diagram of a rooted tree, the root is generally marked distinctly. We will

show the root enclosed in a small triangle. All rooted trees with four vertices are shown in Fig. 3-11. Generally, the term *tree* means trees without any root. However, for emphasis they are sometimes called *free trees* (or *nonrooted trees*) to differentiate them from the rooted kind.

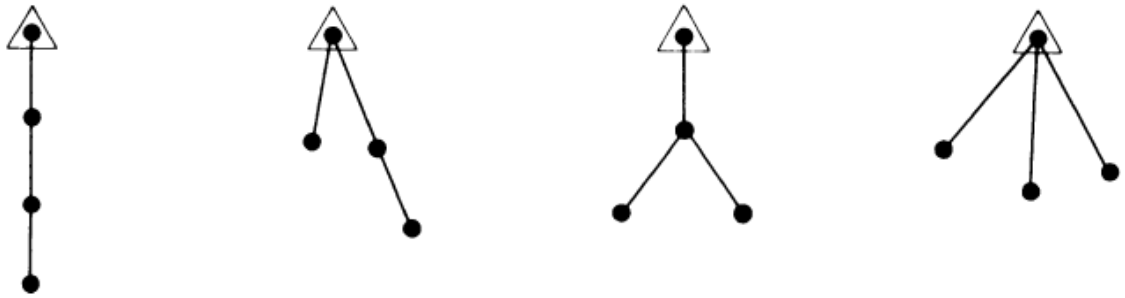


Fig. 3-11 Rooted trees with four vertices.

Binary Trees: A special class of rooted trees, called binary rooted trees, is of particular interest, since they are extensively used in the study of computer search methods, binary identification problems, and variable-length binary codes. A *binary tree* is defined as a tree in which there is exactly one vertex of degree two, and each of the remaining vertices is of degree one or three (Fig. 3-12). (Obviously, we are talking about trees with three or more vertices.) Since the vertex of degree two is distinct from all other vertices, this vertex serves as a root. Thus every binary tree is a rooted tree. Two properties of binary trees follow directly from the definition:

1. The number of vertices n in a binary tree is always odd. This is because there is exactly one vertex of even degree, and the remaining $n - 1$ vertices are of odd degrees. Since from Theorem 1-1 the number of vertices of odd degrees is even, $n - 1$ is even. Hence n is odd.

2. Let p be the number of pendant vertices in a binary tree T. Then $n - p - 1$ is the number of vertices of degree three. Therefore, the number of edges in T equals

$$\frac{1}{2}[p + 3(n - p - 1) + 2] = n - 1;$$

hence

$$p = \frac{n + 1}{2}. \qquad (3\text{-}1)$$

A nonpendant vertex in a tree is called an *internal vertex*. It follows from Eq. (3-1) that the number of internal vertices in a binary tree is one less than the number of pendant vertices. In a binary tree a vertex v_i is said to be at *level* l_i if v_i is at a distance of l_i from the root. Thus the root is at level 0. A 13-vertex, four-level binary tree is shown in Fig. 3-12. The number of vertices at levels 1, 2, 3, and 4 are 2, 2, 4, and 4, respectively.

One of the most straightforward applications of binary trees is in search procedures. Each vertex of a binary tree represents a test with two possible

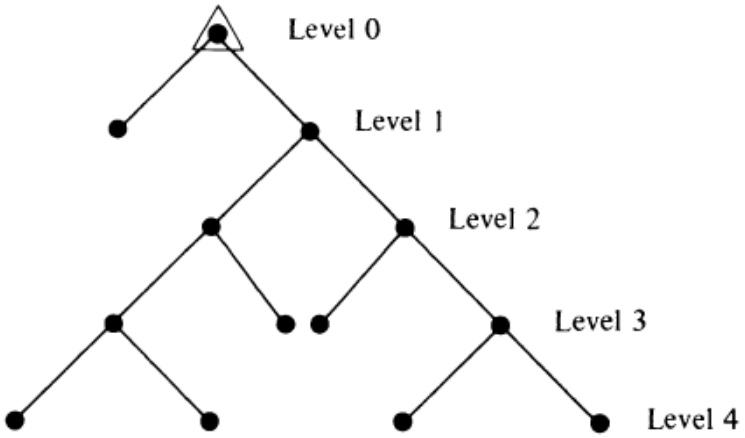


Fig. 3-12 A 13-vertex, 4-level binary tree.

outcomes. We start at the root, and the outcome of the test at the root sends us to one of the two vertices at the next level, where further tests are made, and so on. Reaching a specified pendant vertex (the goal of the search) terminates the search. For such a search procedure it is often important to construct a binary tree in which, for a given number of vertices n, the vertex farthest from the root is as close to the root as possible. Clearly, there can be only one vertex (the root) at level 0, at most two vertices at level 1, at most four vertices at level 2, and so on. Therefore, the maximum number of vertices possible in a k-level binary tree is

$$2^0 + 2^1 + 2^2 + \cdots + 2^k \geq n.$$

The maximum level, $l_{\max}$, of any vertex in a binary tree is called the *height* of the tree. It is easy to see that the minimum possible height of an n-vertex binary tree is

$$2^0 + 2^1 + 2^2 + \cdots + 2^k \geq n.$$

The maximum level, l_{max}, of any vertex in a binary tree is called the *height* of the tree. It is easy to see that the minimum possible height of an n-vertex binary tree is

$$\min l_{max} = \lceil \log_2 (n + 1) - 1 \rceil, \tag{3-2}$$

where $\lceil n \rceil$ denotes the smallest integer greater than or equal to n.

On the other hand, to construct a binary tree for a given n such that the farthest vertex is as far as possible from the root, we must have exactly two vertices at each level, except at the 0 level. Therefore,

$$\max l_{max} = \frac{n - 1}{2}. \tag{3-3}$$

For $n = 11$, binary trees realizing both these extremes are shown in Fig. 3-13.

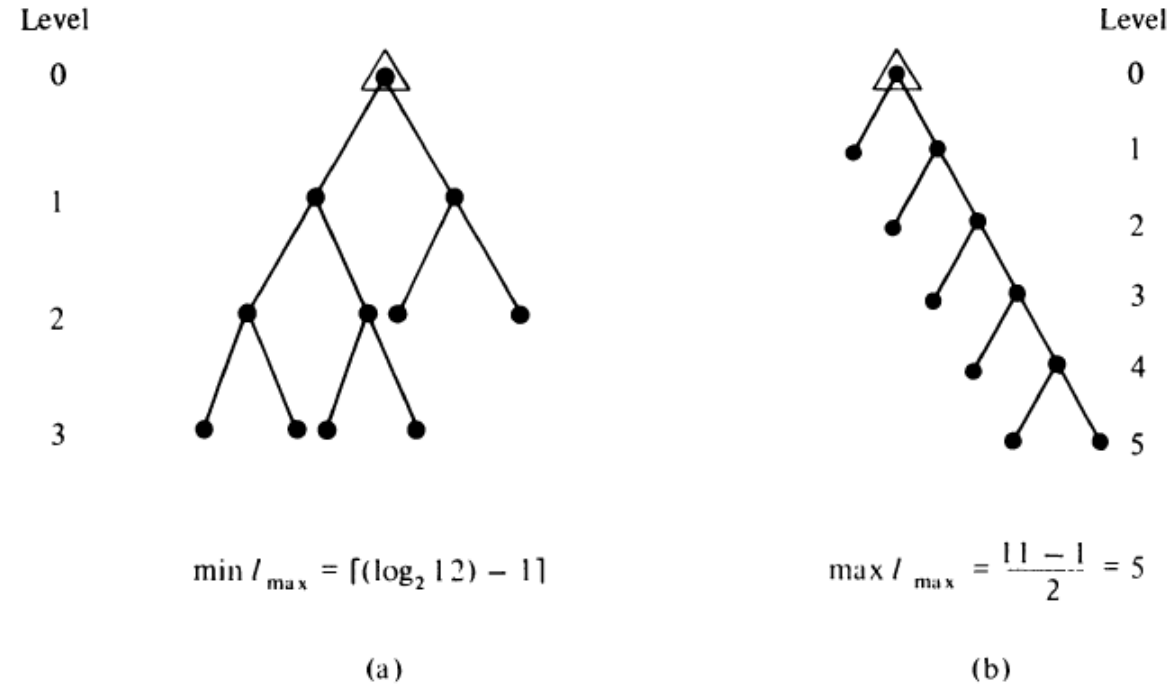


Fig. 3-13 Two 11-vertex binary trees.

In analysis of algorithms we are generally interested in computing the sum of the levels of all pendant vertices. This quantity, known as *the path length* (or external path length) *of a tree*, can be defined as the sum of the path lengths from the root to all pendant vertices. The path length of the binary tree in Fig. 3-12, for example, is

$$1 + 3 + 3 + 4 + 4 + 4 + 4 = 23.$$

The path lengths of trees in Figs. 3-13(a) and (b) are 16 and 20, respectively. The importance of the path length of a tree lies in the fact that this quantity is often directly related to the execution time of an algorithm.

It can be shown that the type of binary tree in Fig. 3-13(a) (i.e., a tree with $2^{l_{max}-1}$ vertices at level $l_{max} - 1$) yields the minimum path length for a given n.

Weighted Path Length: In some applications, every pendant vertex v_j of a binary tree has associated with it a positive real number w_j. Given w_1, $w_2, \ldots, w_m$ the problem is to construct a binary tree (with m pendant vertices) that minimizes

$$\sum w_j l_j,$$

where l_j is the level of pendant vertex v_j, and the sum is taken over all pendant vertices. Let us illustrate the significance of this problem with a simple example.

A Coke machine is to identify, by a sequence of tests, the coin that is put into the machine. Only pennies, nickels, dimes, and quarters can go through the slot. Let us assume that the probabilities of a coin being a penny, a nickel, a dime, and a quarter are .05, .15, .5, and .30, respectively. Each test has the effect of partitioning the four types of coins into two complementary sets and asserting the unknown coin to be in one of the two sets. Thus for four coins we have $2^3 - 1$ such tests. If the time taken for each test is the same, what sequence of tests will minimize the expected time taken by the Coke machine to identify the coin?

The solution requires the construction of a binary tree with four pendant vertices (and therefore three internal vertices) v_1, v_2, v_3, and v_4 and corresponding weights $w_1 = .05$, $w_2 = .15$, $w_3 = .5$, and $w_4 = .3$, such that the quantity $\sum l_i w_i$ is minimized. The solution is given in Fig. 3-14(a), for which the expected time is $1.7t$, where t is the time taken for each test. Contrast this with Fig. 3-14(b), for which the expected time is $2t$. An algorithm for constructing a binary tree with minimum weighted path length can be found in [3-6].

A Coke machine is to identify, by a sequence of tests, the coin that is put into the machine. Only pennies, nickels, dimes, and quarters can go through the slot. Let us assume that the probabilities of a coin being a penny, a nickel, a dime, and a quarter are .05, .15, .5, and .30, respectively. Each test has the effect of partitioning the four types of coins into two complementary sets and asserting the unknown coin to be in one of the two sets. Thus for four coins we have $2^3 - 1$ such tests. If the time taken for each test is the same, what sequence of tests will minimize the expected time taken by the Coke machine to identify the coin?

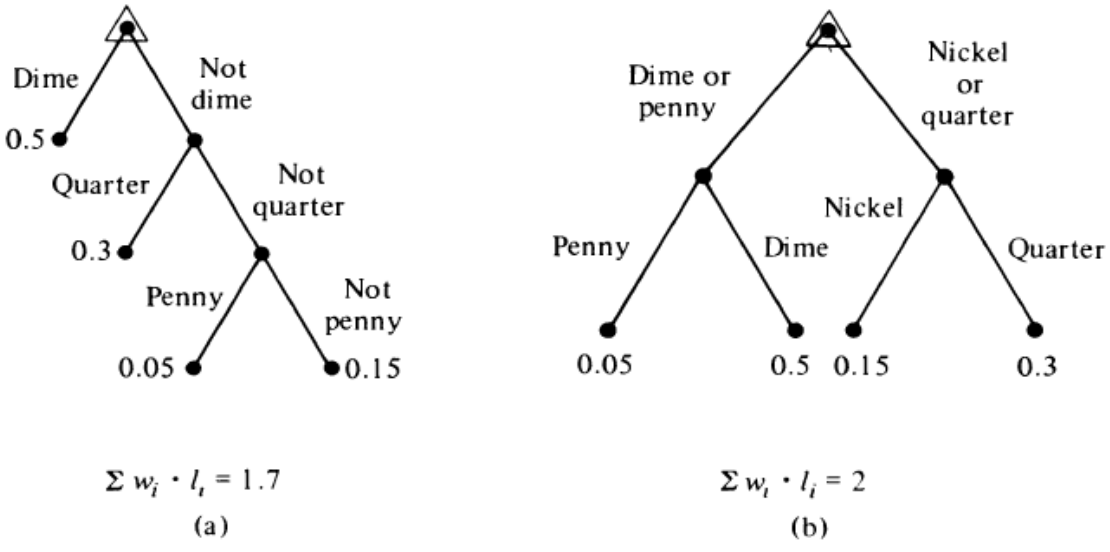
The solution requires the construction of a binary tree with four pendant vertices (and therefore three internal vertices) v_1, v_2, v_3, and v_4 and corresponding weights $w_1 = .05$, $w_2 = .15$, $w_3 = .5$, and $w_4 = .3$, such that the quantity $\sum l_i w_i$ is minimized. The solution is given in Fig. 3-14(a), for which the expected time is $1.7t$, where t is the time taken for each test. Contrast this with Fig. 3-14(b), for which the expected time is $2t$. An algorithm for constructing a binary tree with minimum weighted path length can be found in [3-6].

A Coke machine is to identify, by a sequence of tests, the coin that is put into the machine. Only pennies, nickels, dimes, and quarters can go through the slot. Let us assume that the probabilities of a coin being a penny, a nickel, a dime, and a quarter are .05, .15, .5, and .30, respectively. Each test has the effect of partitioning the four types of coins into two complementary sets and asserting the unknown coin to be in one of the two sets. Thus for four coins we have $2^3 - 1$ such tests. If the time taken for each test is the same, what sequence of tests will minimize the expected time taken by the Coke machine to identify the coin?

The solution requires the construction of a binary tree with four pendant vertices (and therefore three internal vertices) v_1, v_2, v_3, and v_4 and corresponding weights $w_1 = .05$, $w_2 = .15$, $w_3 = .5$, and $w_4 = .3$, such that the quantity $\sum l_i w_i$ is minimized. The solution is given in Fig. 3-14(a), for which the expected time is $1.7t$, where t is the time taken for each test. Contrast this with Fig. 3-14(b), for which the expected time is $2t$. An algorithm for constructing a binary tree with minimum weighted path length can be found in [3-6].

In this problem of a Coke machine, many interesting variations are possible. For example, not all possible tests may be available, or they may not all consume the same time.

Binary trees with minimum weighted path length have also been used in



$\Sigma w_i \cdot l_i = 1.7$ (a)

$\Sigma w_i \cdot l_i = 2$ (b)

Fig. 3-14 Two binary trees with weighted pendant vertices.

constructing variable-length binary codes, where the letters of the alphabet (A, B, C, . . . , Z) are represented by binary digits. Since different letters have different frequencies of occurrence (frequencies are interpreted as weights $w_1, w_2, \ldots, w_{26}$), a binary tree with minimum weighted path length corresponds to a binary code of minimum cost; see [3-6]. For more on minimum-path binary trees and their applications the reader is referred to [3-5] and [3-7].

ON COUNTING TREES

In 1857, Arthur Cayley discovered trees while he was trying to count the number of structural isomers of the saturated hydrocarbons (or paraffin series) C_kH_{2k+2}. He used a connected graph to represent the C_kH_{2k+2} molecule. Corresponding to their chemical valencies, a carbon atom was represented by a vertex of degree four and a hydrogen atom by a vertex of degree one (pendant vertices). The total number of vertices in such a graph is

$$n = 3k + 2,$$

and the total number of edges is

$$e = \frac{1}{2}(\text{sum of degrees}) = \frac{1}{2}(4k + 2k + 2)$$

$$= 3k + 1.$$

Since the graph is connected and the number of edges is one less than the number of vertices, it is a tree. Thus the problem of counting structural isomers of a given hydrocarbon becomes the problem of counting trees (with certain qualifying properties, to be sure).

The first question Cayley asked was: what is the number of different trees that one can construct with n distinct (or labeled) vertices? If $n = 4$, for instance, we have 16 trees, as shown in Fig. 3-15. The reader can satisfy himself that there are no more trees of four vertices. (Of course, some of these trees are isomorphic—to be discussed later.)

A graph in which each vertex is assigned a unique name or label (i.e., no two vertices have the same label), as in Fig. 3-15, is called a *labeled graph*. The distinction between a labeled and an unlabeled graph is very important when we are counting the number of different graphs. For instance, the four

graphs in the first row in Fig. 3-15 are counted as four different trees (even though they are isomorphic) only because the vertices are labeled. If there

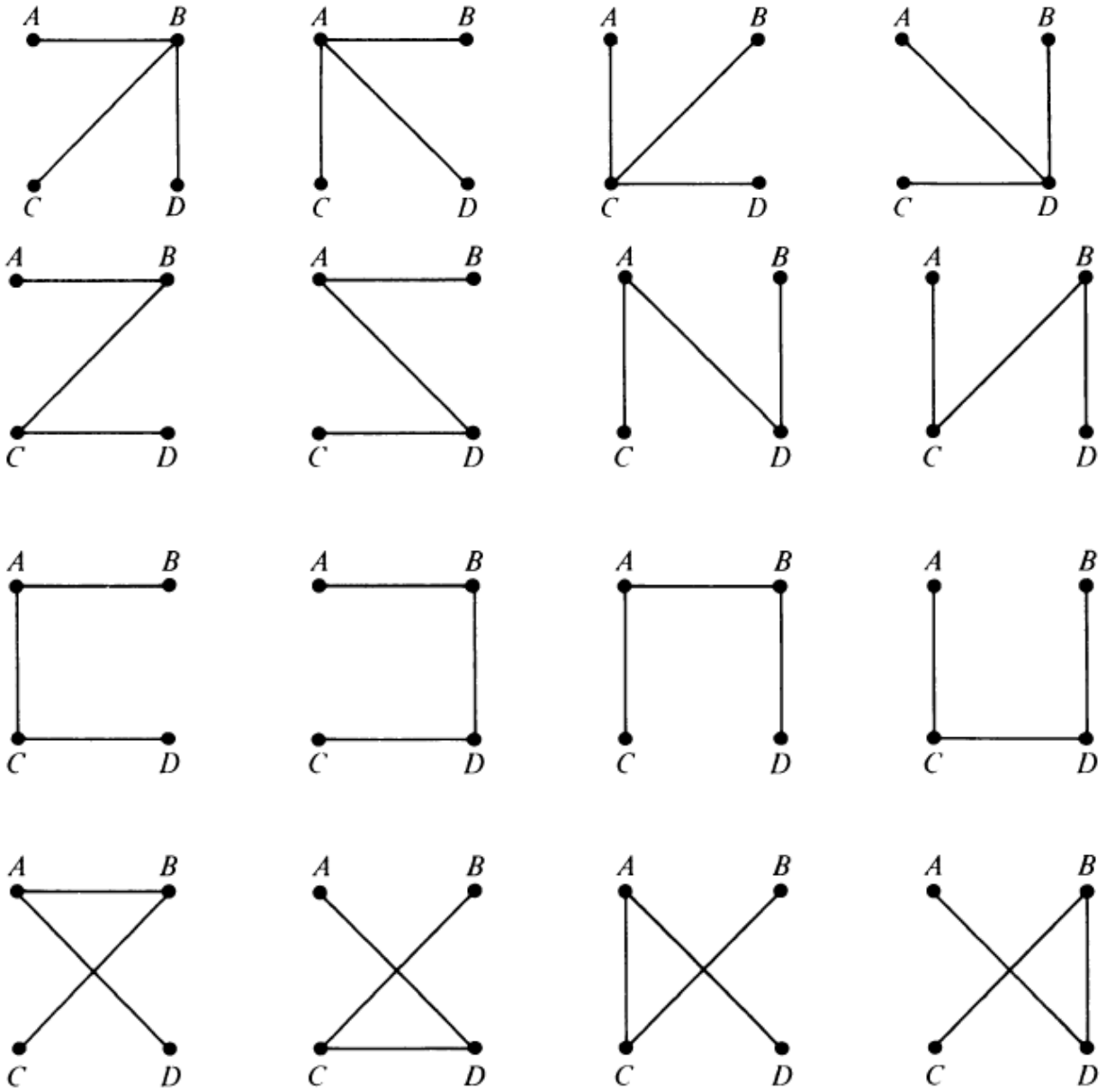


Fig. 3-15 All 16 trees of four labeled vertices.

were no distinction made between A, B, C, or D, these four trees would be counted as one. A careful inspection of the graphs in Fig. 3-15 will reveal that the number of unlabeled trees with four vertices (no distinction made between A, B, C, and D) is only two. But first we shall continue with counting labeled trees.

The following well-known theorem for counting trees was first stated and proved by Cayley, and is therefore called Cayley's theorem.

THEOREM 3-10

The number of labeled trees with n vertices ($n \geq 2$) is n^{n-2}.

Proof: The result was first stated and proved by Cayley. Many different proofs with various approaches (all somewhat involved) have been published since. An excellent summary of 10 such proofs is given by Moon [3-9]. We will give one proof in Chapter 10.

Unlabeled Trees: In the actual counting of isomers of C_kH_{2k+2}, Theorem 3-10 is not enough. In addition to the constraints on the degree of the vertices, two observations should be made:

1. Since the vertices representing hydrogen are pendant, they go with carbon atoms only one way, and hence make no contribution to isomerism. Therefore, we need not show any hydrogen vertices.

2. Thus the tree representing C_kH_{2k+2} reduces to one with k vertices, each representing a carbon atom. In this tree no distinction can be made between vertices, and therefore it is unlabeled.

Thus for butane, C_4H_{10}, there are only two distinct trees (Fig. 3-16). As every organic chemist knows, there are indeed exactly two different types of butanes: *n*-butane and isobutane. It may be noted in passing that the four trees in the first row of Fig. 3-15 are isomorphic to the one in Fig. 3-16(a); and the other 12 are isomorphic to Fig. 3-16(b).

Fig. 3-16 All trees of four unlabeled vertices.

The problem of counting trees of different types will be taken up again and discussed more thoroughly in Chapter 10.

SPANNING TREES

So far we have discussed the tree and its properties when it occurs as a graph by itself. Now we shall study the tree as a subgraph of another graph. A given graph has numerous subgraphs—from e edges, 2^e distinct combinations are possible. Obviously, some of these subgraphs will be trees. Out of these trees we are particularly interested in certain types of trees, called *spanning trees*—as defined next.

A tree T is said to be a *spanning tree* of a connected graph G if T is a subgraph of G and T contains all vertices of G. For instance, the subgraph in heavy lines in Fig. 3-17 is a spanning tree of the graph shown.

Since the vertices of G are barely hanging together in a spanning tree, it is a sort of skeleton of the original graph G. This is why a spanning tree is sometimes referred to as a *skeleton* or *scaffolding* of G. Since spanning trees are the largest (with maximum number of edges) trees among all trees in G, it is also quite appropriate to call a spanning tree a *maximal tree subgraph* or *maximal tree* of G.

It is to be noted that a spanning tree is defined only for a connected graph, because a tree is always connected, and in a disconnected graph of n vertices we cannot find a connected subgraph with n vertices. Each component (which by definition is connected) of a disconnected graph, however, does have a spanning tree. Thus a disconnected graph with k components has a *spanning forest* consisting of k spanning trees. (A collection of trees is called a *forest*.)

Finding a spanning tree of a connected graph G is simple. If G has no circuit, it is its own spanning tree. If G has a circuit, delete an edge from the circuit. This will still leave the graph connected (Problem 2-10). If there are more circuits, repeat the operation till an edge from the last circuit is deleted—leaving a connected, circuit-free graph that contains all the vertices of G. Thus we have

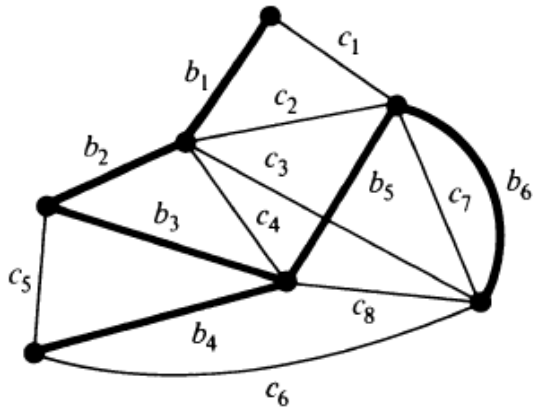


Fig. 3-17 Spanning tree.

THEOREM 3-11

Every connected graph has at least one spanning tree.

An edge in a spanning tree T is called a *branch* of T. An edge of G that is not in a given spanning tree T is called a *chord*. In electrical engineering a chord is sometimes referred to as a *tie* or a *link*. For instance, edges b_1, b_2, b_3, b_4, b_5, and b_6 are branches of the spanning tree shown in Fig. 3-17, while edges c_1, c_2, c_3, c_4, c_5, c_6, c_7, and c_8 are chords. It must be kept in mind that branches and chords are defined only with respect to a given spanning tree. An edge that is a branch of one spanning tree T_1 (in a graph G) may be a chord with respect to another spanning tree T_2.

It is sometimes convenient to consider a connected graph G as a union of two subgraphs, T and $\bar{T}$; that is,

$$T \cup \bar{T} = G,$$

where T is a spanning tree, and $\bar{T}$ is the complement of T in G. Since the subgraph $\bar{T}$ is the collection of chords, it is quite appropriately referred to as the *chord set* (or *tie set* or *cotree*) of T. From the definition, and from Theorem 3-3, the following theorem is evident.

THEOREM 3-12

With respect to any of its spanning trees, a connected graph of n vertices and e edges has $n - 1$ tree branches and $e - n + 1$ chords.

For example, the graph in Fig. 3-17 (with $n = 7$, $e = 14$), has six tree branches and eight chords with respect to the spanning tree $\{b_1, b_2, b_3, b_4, b_5, b_6\}$. Any other spanning tree will yield the same numbers.

If we have an electric network with e elements (edges) and n nodes (vertices), what is the minimum number of elements we must remove to eliminate all circuits in the network? The answer is $e - n + 1$. Similarly, if we have a farm consisting of six walled plots of land, as shown in Fig. 3-18, and these plots are full of water, how many walls will have to be broken so that all the water can be drained out? Here $n = 10$ and $e = 15$. We shall have to select

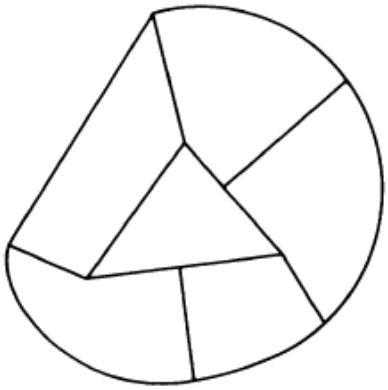


Fig. 3-18 Farm with walled plots of land.

a set of six ($15 - 10 + 1 = 6$) walls such that the remaining nine constitute a spanning tree. Breaking these six walls will drain the water out.

Rank and Nullity: When someone specifies a graph G, the first thing he is most likely to mention is n, the number of vertices in G. Immediately following comes e, the number of edges in G. Then k, the number of components G has. If $k = 1$, G is connected. How are these three numbers of a graph related? Since every component of a graph must have at least one vertex, $n \geq k$. Moreoever, the number of edges in a component can be no less than the number of vertices in that component minus one. Therefore, $e \geq n - k$. Apart from the constraints $n - k \geq 0$ and $e - n + k \geq 0$, these three numbers n, e, and k are independent, and they are fundamental numbers in graphs. (Needless to mention, these numbers alone are not enough to specify a graph, except for trivial cases.)

From these three numbers are derived two other important numbers called *rank* and *nullity*, defined as

$$\text{rank} \qquad r = n - k,$$

$$\text{nullity} \qquad \mu = e - n + k.$$

The rank of a connected graph is $n - 1$, and the nullity, $e - n + 1$. Although the real significance of these numbers will be clear in Chapter 7, it may be observed here that

$$\text{rank of } G = \text{number of branches in any spanning tree (or forest) of } G,$$

$$\text{nullity of } G = \text{number of chords in } G,$$

$$\text{rank} + \text{nullity} = \text{number of edges in } G.$$

The nullity of a graph is also referred to as its *cyclomatic number*, or *first Betti number*.

FUNDAMENTAL CIRCUITS

THEOREM 3-13

A connected graph G is a tree if and only if adding an edge between any two vertices in G creates exactly one circuit.

Let us now consider a spanning tree T in a connected graph G. Adding any one chord to T will create exactly one circuit. Such a circuit, formed by adding a chord to a spanning tree, is called a *fundamental circuit*.

How many fundamental circuits does a graph have? Exactly as many as the number of chords, $\mu\ (= e - n + k)$. How many circuits does a graph have in all? We know that one circuit is created by adding any one chord to a tree. Suppose that we add one more chord. Will it create exactly one more circuit? What happens if we add all the chords simultaneously to the tree?

Let us look at the tree $\{b_1, b_2, b_3, b_4, b_5, b_6\}$ in Fig. 3-17. Adding c_1 to it, we get a subgraph $\{b_1, b_2, b_3, b_4, b_5, b_6, c_1\}$, which has one circuit (fundamental circuit), $\{b_1, b_2, b_3, b_5, c_1\}$. Had we added the chord c_2 (instead of c_1) to the tree, we would have obtained a different fundamental circuit, $\{b_2, b_3, b_5, c_2\}$. Now suppose that we add both chords c_1 and c_2 to the tree. The subgraph $\{b_1, b_2, b_3, b_4, b_5, b_6, c_1, c_2\}$ has not only the fundamental circuits we just mentioned, but it has also a third circuit, $\{b_1, c_1, c_2\}$, which is not a fundamental circuit. Although there are 75 circuits in Fig. 3-17 (enumerated by computer), only eight are fundamental circuits, each formed by one chord (together with the tree branches).

Two comments may be appropriate here. First, a circuit is a fundamental circuit only with respect to a given spanning tree. A given circuit may be fundamental with respect to one spanning tree, but not with respect to a different spanning tree of the same graph. Although the number of fundamental circuits (as well as the total number of circuits) in a graph is fixed, the circuits that become fundamental change with the spanning trees.

Second, in most applications we are not interested in all the circuits of a graph, but only in a set of fundamental circuits, which fortuitously are a lot easier to track. The concept of a fundamental circuit, introduced by Kirchhoff, is of enormous significance in electrical network analysis. What Kirchhoff showed, which now every sophomore in electrical engineering knows, is that no matter how many circuits a network contains we need consider only fundamental circuits with respect to any spanning tree. The rest of the circuits (as we shall prove rigorously in Chapter 7) are combinations of some fundamental circuits.

Possible Questions

2 Mark Questions:

1. Define divisible with example.
2. Prove that if $a|b$ and $a|c$,then $a|(bx+cy)$ for arbitrary integers x and y.
3. Define greatest common divisor with example.
4. What is relatively prime.
5. Discuss about Diophantine equation.
6. Prove that if p is a prime and $p|ab$, then $p|a$ or $p|b$.
7. State Euclid theorem.
8. Define Linear congruence.
9. Prove if $\gcd(a,n)=1$, then the linear congruence $ax \equiv b(\mathrm{mod}\, n)$ has a unique solution modulo n.
10. State Chinese Remainder theorem.

8 Mark Questions:

1. Prove that the linear Diophantine equation $ax+by=c$ has a solution if and only if $d|c$,where $d=\gcd(a,b)$. If x_0, y_0 is any particular solution of this equation then all other solutions are given by

$$x = x_0 + (b/d)t, \quad y = y_0 - (a/d)t$$

for varying integers t.

2. Determine all the solutions in the integers of each of the following Diophantine equations:
   a) $56x+72y=40$;
   b) $24x+138y=18$;
   c) $221x+91y=117$;
   d) $84x-438y=156$.

3. Determine all the solutions in the Positive integers of each of the following Diophantine equations:

   a) $30x+17y=300$;

   b) $54x + 21y = 906$;

   c) $123x+360y=99$;

4. State and prove fundamental theorem of Arithmetic.

5. State and prove Euclid Lemma.

6. Prove that if p_n is the n[th] prime number, then $p_n \le 2^{2^{n-1}}$.

7. Prove that there are infinite number of primes of the form $4n+3$.

8. Prove that the linear congruence $ax \equiv b(\text{mod } n)$ has a solution if and only if $d|b$, where $d = \gcd(a,n)$. if $d|b$, then it has d mutually in-congruent solutions modulo n.

9. State and Prove Chinese Remainder theorem.

10. Solve the following linear congruence:

    a) $25x \equiv 15(\text{mod } 29)$ b) $5x \equiv 2(\text{mod } 26)$

UNIT-II

SYLLABUS

Fermat's Little theorem, Wilson's theorem. Number theoretic functions, sum and number of divisors, Totally multiplicative functions , Definition and properties of the Dirichlet product.

FINDING ALL SPANNING TREES OF A GRAPH

Usually, in a given connected graph there are a large number of spanning trees. In many applications we require all spanning trees. One reasonable way to generate spanning trees of a graph is to start with a given spanning tree, say tree T_1 ($a\ b\ c\ d$ in Fig. 3-19). Add a chord, say h, to the tree T_1. This forms a fundamental circuit ($b\ c\ h\ d$ in Fig. 3-19). Removal of any branch, say c, from the fundamental circuit $b\ c\ h\ d$ just formed will create a new

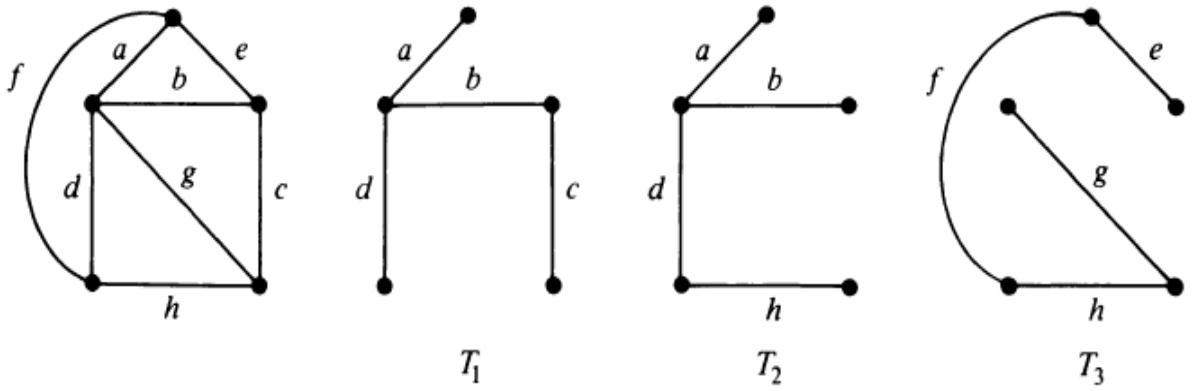


Fig. 3-19 Graph and three of its spanning trees.

spanning tree T_2. This generation of one spanning tree from another, through addition of a chord and deletion of an appropriate branch, is called a *cyclic interchange* or *elementary tree transformation*. (Such a transformation is a standard operation in the iteration sequence for solving certain transportation problems.)

In the above procedure, instead of deleting branch c, we could have deleted d or b and thus would have obtained two additional spanning trees $a\ b\ c\ h$ and $a\ c\ h\ d$. Moreover, after generating these three trees, each with chord h in it, we can restart with T_1 and add a different chord (e, f, or g) and repeat the process of obtaining a different spanning tree each time a branch is deleted from the fundamental circuit formed. Thus we have a procedure for generating spanning trees for any given graph.

As we shall see in Chapter 13, the topological analysis of a linear electrical network essentially reduces to the generation of trees in the corresponding graph. Therefore, finding an efficient procedure for generating all trees of a graph is a very important practical problem.

The procedure outlined above raises many questions. Can we start from any spanning tree and get a desired spanning tree by a number of *cyclic exchanges*? Can we get all spanning trees of a given graph in this fashion? How long will we have to continue exchanging edges? Out of all possible spanning trees that we can start with, is there a preferred one for starting? Let us try to answer some of these questions; others will have to wait until Chapters 7, 10, and 11.

The distance between two spanning trees T_i and T_j of a graph G is defined as the number of edges of G present in one tree but not in the other. This distance may be written as $d(T_i, T_j)$. For instance, in Fig. 3-19 $d(T_2, T_3) = 3$.

Let $T_i \oplus T_j$ be the ring sum of two spanning trees T_i and T_j of G (as defined in Chapter 2, $T_i \oplus T_j$ is the subgraph of G containing all edges of G that are either in T_i or in T_j but not in both). Let $N(g)$ denote the number of edges in a graph g. Then, from definition,

$$d(T_i, T_j) = \frac{1}{2} N(T_i \oplus T_j).$$

It is not difficult to see that the number $d(T_i, T_j)$ is the minimum number of cyclic interchanges involved in going from T_i to T_j. The reader is encouraged to prove the following two theorems.

THEOREM 3-14

The distance between the spanning trees of a graph is a *metric*. That is, it satisfies

$$d(T_i, T_j) \geq 0 \quad \text{and} \quad d(T_i, T_j) = 0 \text{ if and only if } T_i = T_j,$$
$$d(T_i, T_j) = d(T_j, T_i),$$
$$d(T_i, T_j) \leq d(T_i, T_k) + d(T_k, T_j).$$

THEOREM 3-15

Starting from any spanning tree of a graph G, we can obtain every spanning tree of G by successive cyclic exchanges.

Since in a connected graph G of rank r (i.e., of $r + 1$ vertices) a spanning tree has r edges, we have the following results:

The maximum distance between any two spanning trees in G is

$$\max d(T_i, T_j) = \frac{1}{2} \max N(T_i \oplus T_j)$$

$$\leq r, \text{ the rank of } G.$$

Also, if μ is the nullity of G, we know that no more than μ edges of a spanning tree T_i can be replaced to get another tree T_j.

Hence $$\max d(T_i, T_j) \leq \mu;$$

combining the two,

$$\max d(T_i, T_j) \leq \min(\mu, r),$$

where $\min(\mu, r)$ is the smaller of the two numbers μ and r of the graph G.

Central Tree: For a spanning tree T_0 of a graph G, let $\max_i d(T_0, T_i)$ denote the maximal distance between T_0 and any other spanning tree of G. Then T_0 is called a *central tree* of G if

$$\max_i d(T_0, T_i) \leq \max_j d(T, T_j) \qquad \text{for every tree } T \text{ of } G.$$

The concept of a central tree is useful in enumerating all trees of a given graph. A central tree in a graph is, in general, not unique. For more on central trees the reader should see [3-1] and [3-4].

Tree Graph: The *tree graph* of a given graph G is defined as a graph in which each vertex corresponds to a spanning tree of G, and each edge corresponds to a *cyclic interchange* between the spanning trees of G represented by the two end vertices of the edge. From Theorem 3-15 we know that starting from any spanning tree we can obtain all other spanning trees through cyclic interchanges (or elementary tree transformations). Therefore, the tree

graph of any given (finite, connected) graph is connected. For additional properties of tree graphs, the reader should see [3-3].

SPANNING TREES IN A WEIGHTED GRAPH

As discussed earlier in this chapter, a spanning tree in a graph G is a minimal subgraph connecting all the vertices of G. If graph G is a weighted graph (i.e., if there is a real number associated with each edge of G), then the *weight of a spanning tree* T of G is defined as the sum of the weights of all the branches in T. In general, different spanning trees of G will have different weights. Among all the spanning trees of G, one with the *smallest* weight is of practical significance. (There may be several spanning trees with the smallest weight; for instance, in a graph of n vertices in which every edge has unit weight, all spanning trees have a weight of $n - 1$ units.) A spanning tree with the smallest weight in a weighted graph is called a *shortest spanning tree* or *shortest-distance spanning tree* or *minimal spanning tree.*

One possible application of the shortest spanning tree is as follows: Suppose that we are to connect n cities $v_1, v_2, \ldots, v_n$ through a network of roads. The cost c_{ij} of building a direct road between v_i and v_j is given for all pairs of cities where roads can be built. (There may be pairs of cities between which no direct road can be built.) The problem is then to find the least expensive network that connects all n cities together. It is immediately evident that this connected network must be a tree: otherwise, we can always remove some edges and get a connected graph with smaller weight. Thus the problem of connecting n cities with a least expensive network is the problem of finding a shortest spanning tree in a connected weighted graph of n vertices. A necessary and sufficient condition for a spanning tree to be shortest is given by

THEOREM 3-16

A spanning tree T (of a given weighted connected graph G) is a shortest spanning tree (of G) if and only if there exists no other spanning tree (of G) at a distance of one from T whose weight is smaller than that of T.

Proof: The necessary or the "only if" condition is obvious; otherwise, we shall get another tree shorter than T by a cyclic interchange. The fact that this condition is also sufficient is remarkable and is not obvious. It can be proved as follows:

Let T_1 be a spanning tree in G satisfying the hypothesis of the theorem (i.e., there is no spanning tree at a distance of one from T_1 which is shorter than T_1).

The proof will be completed by showing that if T_2 is a shortest spanning tree (different from T_1) in G, the weight of T_1 will also be equal to that of T_2. Let T_2 be a shortest spanning tree in G. Clearly, T_2 must also satisfy the hypothesis of the theorem (otherwise there will be a spanning tree shorter than T_2 at a distance of one from T_2, violating the assumption that T_2 is shortest).

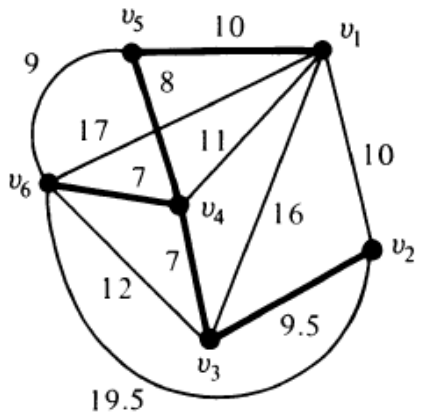
Consider an edge e in T_2 which is not in T_1. Adding e to T_1 forms a fundamental circuit with branches in T_1. Some, but not all, of the branches in T_1 that form the fundamental circuit with e may also be in T_2; each of these branches in T_1 has a weight smaller than or equal to that of e, because of the assumption on T_1. Amongst all those edges in this circuit which are not in T_2 at least one, say b_j, must form some fundamental circuit (with respect to T_2) containing e. Because of the minimality assumption on T_2 weight of b_j cannot be less than that of e. Therefore b_j must have the same weight as e. Hence the spanning tree $T_1' = (T_1 \cup e - b_j)$, obtained from T_1 through one cycle exchange, has the same weight as T_1. But T_1 has one edge more in common with T_2, and it satisfies the condition of Theorem 3-16. This argument can be repeated, producing a series of trees of equal weight, $T_1, T_1, T_1, \ldots$, each a unit distance closer to T_2, until we get T_2 itself.

This proves that if none of the spanning trees at a unit distance from T is shorter than T, no spanning tree shorter than T exists in the graph. ■

Algorithm for Shortest Spanning Tree: There are several methods available for actually finding a shortest spanning tree in a given graph, both by hand and by computer. One algorithm due to Kruskal [3-8] is as follows: List all edges of the graph G in order of nondecreasing weight. Next, select a smallest edge of G. Then for each successive step select (from all remaining edges of G) another smallest edge that makes no circuit with the previously selected edges. Continue until $n - 1$ edges have been selected, and these edges will constitute the desired shortest spanning tree. The validity of the method follows from Theorem 3-16.

Another algorithm, which does not require listing all edges in order of nondecreasing weight or checking at each step if a newly selected edge forms a circuit, is due to Prim [3-10]. For Prim's algorithm, draw n isolated vertices and label them $v_1, v_2, \ldots, v_n$. Tabulate the given weights of the edges of G in an n by n table. (Note that the entries in the table are symmetric with respect to the diagonal, and the diagonal is empty.) Set the weights of nonexistent edges (corresponding to those pairs of cities between which no direct road can be built) as very large.

Start from vertex v_1 and connect it to its nearest neighbor (i.e., to the vertex which has the smallest entry in row 1 of the table), say v_k. Now consider v_1 and v_k as one subgraph, and connect this subgraph to its closest neighbor (i.e., to a vertex other than v_1 and v_k that has the smallest entry among all entries in rows 1 and k). Let this new vertex be v_i. Next regard the tree with vertices v_1, v_k, and v_i as one subgraph, and continue the process until all n vertices have been connected by $n - 1$ edges. Let us now illustrate this method of finding a shortest spanning tree.



(a)

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	–	10	16	11	10	17
v_2	10	–	9.5	∞	∞	19.5
v_3	16	9 5	–	7	∞	12
v_4	11	∞	7	–	8	7
v_5	10	∞	∞	8	–	9
v_6	17	19.5	12	7	9	–

(b)

Fig. 3-20 Shortest spanning tree in a weighted graph.

A connected weighted graph with 6 vertices and 12 edges is shown in Fig. 3-20(a). The weight of its edges is tabulated in Fig. 3-20(b). We start with v_1 and pick the smallest entry in row 1, which is either (v_1, v_2) or (v_1, v_5). Let us pick (v_1, v_5). [Had we picked (v_1, v_2) we would have obtained a different shortest tree with the same weight.] The closest neighbor of subgraph (v_1, v_5) is v_4, as can be seen by examining all the entries in rows 1 and 5. The three remaining edges selected following the above procedure turn out to be (v_4, v_6), (v_4, v_3), and (v_3, v_2) in that sequence. The resulting tree—a shortest spanning tree—is shown in Fig. 3-20(a) in heavy lines. The weight of this tree is 41.5 units.

Degree-Constrained Shortest Spanning Tree: In a shortest spanning tree resulting from the preceding construction, a vertex v_i can end up with any degree; that is, $1 \leq d(v_i) \leq n - 1$. In some practical cases an upper limit on the degree of every vertex (of the resulting spanning tree) has to be imposed.

For instance, in an electrical wiring problem, one may be required to wire together n pins (using as little wire as possible) with no more than three wires wrapped around any individual pin. Thus, in this particular case,

$$d(v_i) \leq 3 \qquad \text{for every } v_i.$$

Such a spanning tree is called a *degree-constrained shortest spanning tree*.

In general, the problem may be stated as follows: Given a weighted connected graph G, find a shortest spanning tree T in G such that

$$d(v_i) \leq k \qquad \text{for every vertex } v_i \text{ in } T.$$

If $k = 2$, this problem, in fact, reduces to the problem of finding the shortest Hamiltonian path, as well as the traveling-salesman problem (without the salesman returning to his home base), discussed at the end of Chapter 2. So far, no efficient method of finding an arbitrarily degree-constrained shortest spanning tree has been found.

CUT-SETS

In a connected graph G, a *cut-set* is a set of edges† whose removal from G leaves G disconnected, provided removal of no proper subset of these edges disconnects G. For instance, in Fig. 4-1 the set of edges $\{a, c, d, f\}$ is a cut-set. There are many other cut-sets, such as $\{a, b, g\}$, $\{a, b, e, f\}$, and $\{d, h, f\}$. Edge $\{k\}$ alone is also a cut-set. The set of edges $\{a, c, h, d\}$, on the other hand, is *not* a cut-set, because one of its proper subsets, $\{a, c, h\}$, is a cut-set.

To emphasize the fact that no proper subset of a cut-set can be a cut-set, some authors refer to a cut-set as a *minimal cut-set*, a *proper cut-set*, or a *simple cut-set*. Sometimes a cut-set is also called a *cocycle*. We shall just use the term *cut-set*.

A cut-set always "cuts" a graph into two. Therefore, a cut-set can also be defined as a minimal set of edges in a connected graph whose removal reduces the rank of the graph by one. The rank of the graph in Fig. 4.1(b), for in-

†Since a set of edges (together with their end vertices) constitutes a subgraph, a cut-set in G is a subgraph of G.

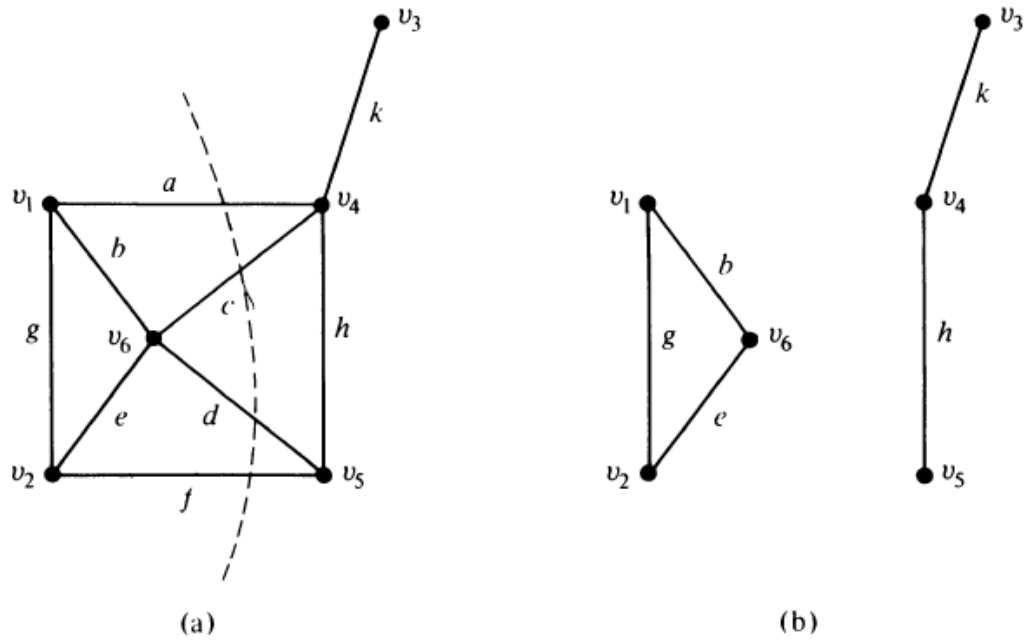


Fig. 4-1 Removal of a cut-set $\{a, c, d, f\}$ from a graph "cuts" it into two.

stance, is four, one less than that of the graph in Fig. 4.1(a). Another way of looking at a cut-set is this: if we partition all the vertices of a connected graph G into two mutually exclusive subsets, a cut-set is a minimal number of edges whose removal from G destroys all paths between these two sets of vertices. For example, in Fig. 4-1(a) cut-set $\{a, c, d, f\}$ connects vertex set $\{v_1, v_2, v_6\}$ with $\{v_3, v_4, v_5\}$. (Note that one or both of these two subsets of vertices may consist of just one vertex.) Since removal of any edge from a tree breaks the tree into two parts, *every edge of a tree is a cut-set*.

Cut-sets are of great importance in studying properties of communication and transportation networks. Suppose, for example, that the six vertices in Fig. 4-1(a) represent six cities connected by telephone lines (edges). We wish to find out if there are any weak spots in the network that need strengthening by means of additional telephone lines. We look at all cut-sets of the graph, and the one with the smallest number of edges is the most vulnerable. In Fig. 4-1(a), the city represented by vertex v_3 can be severed from the rest of the network by the destruction of just one edge. After some additional study of the properties of cut-sets, we shall return to their applications.

SOME PROPERTIES OF A CUT-SET

Consider a spanning tree T in a connected graph G and an arbitrary cut-set S in G. Is it possible for S not to have any edge in common with T? The answer is *no*. Otherwise, removal of the cut-set S from G would not disconnect the graph. Therefore,

THEOREM 4-1

Every cut-set in a connected graph G must contain at least one branch of every spanning tree of G.

Will the converse also be true? In other words, will any minimal set of edges containing at least one branch of every spanning tree be a cut-set? The answer is *yes*, by the following reasoning:

In a given connected graph G, let Q be a minimal set of edges containing at least one branch of every spanning tree of G. Consider $G - Q$, the subgraph that remains after removing the edges in Q from G. Since the subgraph $G - Q$ contains no spanning tree of G, $G - Q$ is disconnected (one component of which may just consist of an isolated vertex). Also, since Q is a minimal set of edges with this property, any edge e from Q returned to $G - Q$ will create at least one spanning tree. Thus the subgraph $G - Q + e$ will be a connected graph. Therefore, Q is a minimal set of edges whose removal from G disconnects G. This, by definition, is a cut-set. Hence

THEOREM 4-2

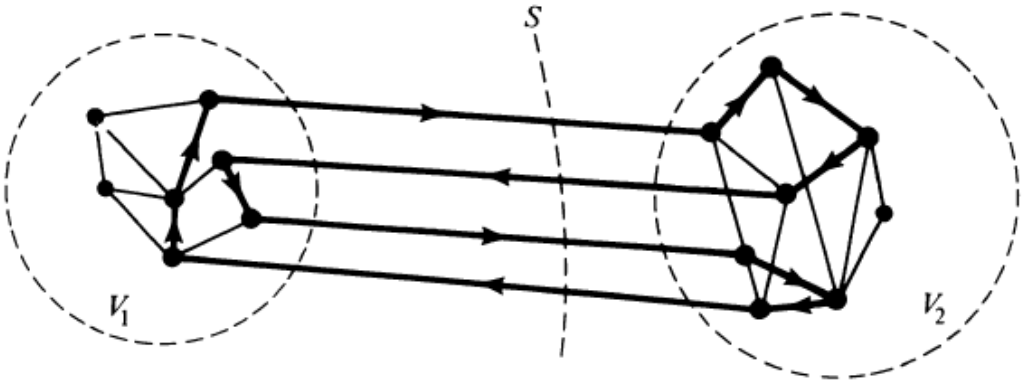
In a connected graph G, any minimal set of edges containing at least one branch of every spanning tree of G is a cut-set.

THEOREM 4-3

Every circuit has an even number of edges in common with any cut-set.

Proof: Consider a cut-set S in graph G (Fig. 4-2). Let the removal of S partition the vertices of G into two (mutually exclusive or disjoint) subsets V_1 and V_2. Consider a circuit Γ in G. If all the vertices in Γ are entirely within vertex set V_1 (or V_2), the number of edges common to S and Γ is zero; that is, $N(S \cap \Gamma) = 0$, an even number.†

If, on the other hand, some vertices in Γ are in V_1 and some in V_2, we traverse



Circuit Γ shown in heavy lines, and is traversed along the direction of the arrows

Fig. 4-2 Circuit and a cut-set in G.

back and forth between the sets V_1 and V_2 as we traverse the circuit (see Fig. 4-2). Because of the closed nature of a circuit, the number of edges we traverse between V_1 and V_2 must be even. And since very edge in S has one end in V_1 and the other in V_2, and no other edge in G has this property (of separating sets V_1 and V_2), the number of edges common to S and Γ is even. ■

ALL CUT-SETS IN A GRAPH

Fundamental Cut-Sets: Consider a spanning tree T of a connected graph G. Take any branch b in T. Since $\{b\}$ is a cut-set in T, $\{b\}$ partitions all vertices of T into two disjoint sets—one at each end of b. Consider the same partition of vertices in G, and the cut set S in G that corresponds to this partition. Cut-set S will contain only one branch b of T, and the rest (if any) of the edges in S are chords with respect to T. Such a cut-set S containing exactly one branch of a tree T is called a *fundamental cut-set* with respect to T. Sometimes a fundamental cut-set is also called a *basic cut-set*. In Fig. 4-3, a spanning tree

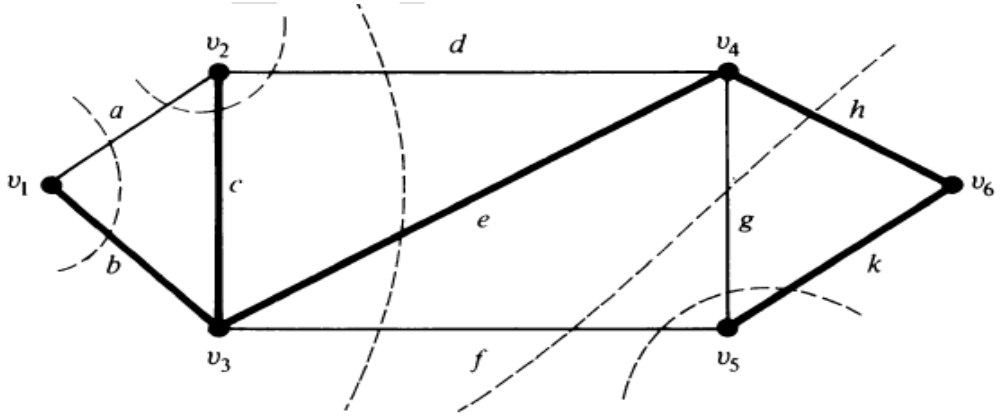


Fig. 4-3 Fundamental cut-sets of a graph.

T (in heavy lines) and all five of the fundamental cut-sets with respect to T are shown (broken lines "cutting" through each cut-set).

Just as every chord of a spanning tree defines a *unique* fundamental circuit, every branch of a spanning tree defines a *unique* fundamental cut-set. It must also be kept in mind that the term fundamental cut-set (like the term fundamental circuit) has meaning only with respect to a *given* spanning tree.

Now we shall show how other cut-sets of a graph can be obtained from a given set of cut-sets.

THEOREM 4-4

The ring sum of any two cut-sets in a graph is either a third cut-set or an edge-disjoint union of cut-sets.

Outline of Proof: Let S_1 and S_2 be two cut-sets in a given connected graph G. Let V_1 and V_2 be the (unique and disjoint) partitioning of the vertex set V of G corresponding to S_1. Let V_3 and V_4 be the partitioning corresponding to S_2. Clearly [see Figs. 4-4(a) and (b)],

$$V_1 \cup V_2 = V \quad \text{and} \quad V_1 \cap V_2 = \varnothing,$$
$$V_3 \cup V_4 = V \quad \text{and} \quad V_3 \cap V_4 = \varnothing.$$

Now let the subset $(V_1 \cap V_4) \cup (V_2 \cap V_3)$ be called V_5, and this by definition is the same as the ring sum $V_1 \oplus V_3$. Similarly, let the subset $(V_1 \cap V_3) \cup (V_2 \cap V_4)$ be called V_6, which is the same as $V_2 \oplus V_3$. See Fig. 4-4(c).

The ring sum of the two cut-sets $S_1 \oplus S_2$ can be seen to consist only of edges that join vertices in V_5 to those in V_6. Also, there are no edges outside $S_1 \oplus S_2$ that join vertices in V_5 to those in V_6.

Thus the set of edges $S_1 \oplus S_2$ produces a partitioning of V into V_5 and V_6 such that

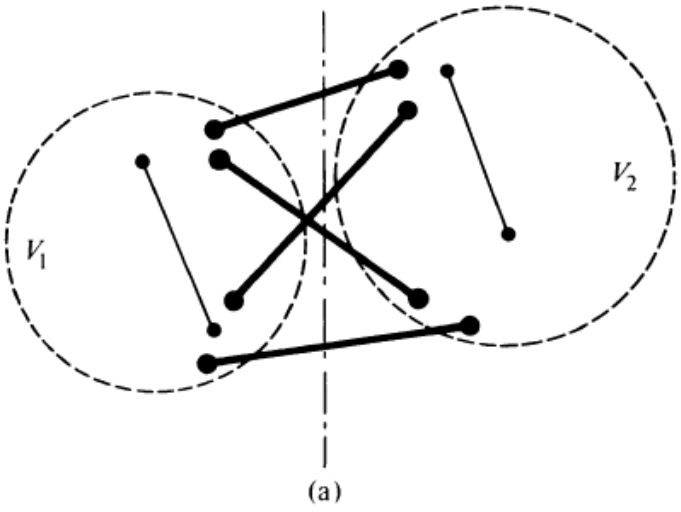
$$V_5 \cup V_6 = V \quad \text{and} \quad V_5 \cap V_6 = \varnothing.$$

Hence $S_1 \oplus S_2$ is a cut-set if the subgraphs containing V_5 and V_6 each remain connected after $S_1 \oplus S_2$ is removed from G. Otherwise, $S_1 \oplus S_2$ is an edge-disjoint union of cut-sets.

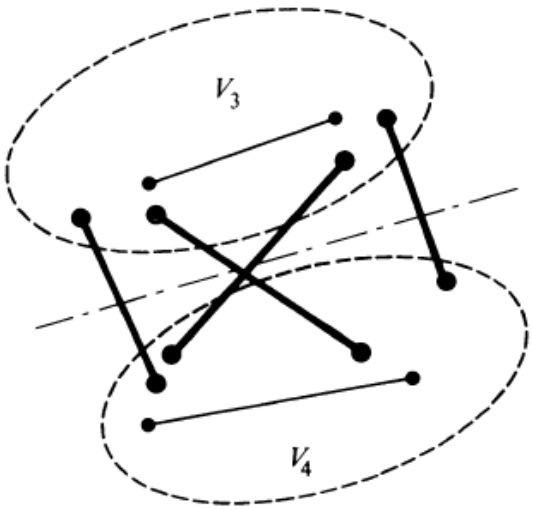
Example: In Fig. 4-3 let us consider ring sums of the following three pairs of cut-sets.

Example: In Fig. 4-3 let us consider ring sums of the following three pairs of cut-sets.

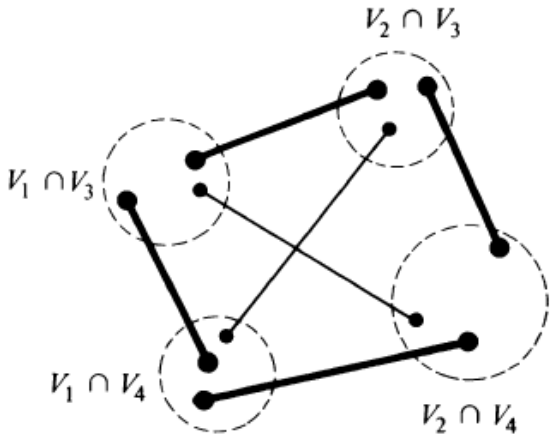
$$\{d, e, f\} \oplus \{f, g, h\} = \{d, e, g, h\}, \quad \text{another cut-set,}$$
$$\{a, b\} \oplus \{b, c, e, f\} = \{a, c, e, f\}, \quad \text{another cut-set,}$$
$$\{d, e, g, h\} \oplus \{f, g, k\} = \{d, e, f, h, k\}$$
$$= \{d, e, f\} \cup \{h, k\}, \text{ an edge-disjoint union of cut-sets.} \blacksquare$$



(a)



(b)



(c)

Fig. 4-4 Two cut-sets and their partitionings.

So we have a method of generating additional cut-sets from a number of given cut-sets. Obviously, we cannot start with any two cut-sets in a given graph and hope to obtain all its cut-sets by this method. What then is a minimal set of cut-sets from which we can obtain every cut-set of G by taking ring sums? The answer (to be proved in Chapter 6) is the set of all fundamental cut-sets with respect to a given spanning tree.

FUNDAMENTAL CIRCUITS AND CUT-SETS

Consider a spanning tree T in a given connected graph G. Let c_i be a chord with respect to T, and let the fundamental circuit made by c_i be called Γ, consisting of k branches $b_1, b_2, \ldots, b_k$ in addition to the chord c_i; that is,

$\Gamma = \{c_i, b_1, b_2, \ldots, b_k\}$ is a fundamental circuit with respect to T.

Every branch of any spanning tree has a fundamental cut-set associated with it. Let S_1 be the fundamental cut-set associated with b_1, consisting of q chords in addition to the branch b_1; that is,

$S_1 = \{b_1, c_1, c_2, \ldots, c_q\}$ is a fundamental cut-set with respect to T.

Because of Theorem 4-3, there must be an even number of edges common to Γ and S_1. Edge b_1 is in both Γ and S_1, and there is only one other edge in Γ (which is c_i) that can possibly also be in S_1. Therefore, we must have two edges b_1 and c_i common to S_1 and Γ. Thus the chord c_i is one of the chords $c_1, c_2, \ldots, c_q$.

Exactly the same argument holds for fundamental cut-sets associated with $b_2, b_3, \ldots,$ and b_k. Therefore, the chord c_i is contained in every fundamental cut-set associated with branches in Γ.

Is it possible for the chord c_i to be in any other fundamental cut-set S' (with respect to T, of course) besides those associated with $b_1, b_2, \ldots$ and b_k? The answer is *no*. Otherwise (since none of the branches in Γ are in S'), there would be only one edge c_i common to S' and Γ, a contradiction to Theorem 4-3. Thus we have an important result.

THEOREM 4-5

With respect to a given spanning tree T, a chord c_i that determines a fundamental circuit Γ occurs in every fundamental cut-set associated with the branches in Γ and in no other.

As an example, consider the spanning tree $\{b, c, e, h, k\}$, shown in heavy lines, in Fig. 4-3. The fundamental circuit made by chord f is

$$\{f, e, h, k\}.$$

The three fundamental cut-sets determined by the three branches e, h, and k are

$$\begin{aligned} &\text{determined by branch } e: \quad \{d, e, f\}, \\ &\text{determined by branch } h: \quad \{f, g, h\}, \\ &\text{determined by branch } k: \quad \{f, g, k\}. \end{aligned}$$

Chord f occurs in each of these three fundamental cut-sets, and there is no other fundamental cut-set that contains f. The converse of Theorem 4-5 is also true.

THEOREM 4-6

With respect to a given spanning tree T, a branch b_i that determines a fundamental cut-set S is contained in every fundamental circuit associated with the chords in S, and in no others.

As an example, consider the spanning tree $\{b, c, e, h, k\}$, shown in heavy lines, in Fig. 4-3. The fundamental circuit made by chord f is

$$\{f, e, h, k\}.$$

The three fundamental cut-sets determined by the three branches e, h, and k are

$$\begin{aligned} &\text{determined by branch } e: \quad \{d, e, f\}, \\ &\text{determined by branch } h: \quad \{f, g, h\}, \\ &\text{determined by branch } k: \quad \{f, g, k\}. \end{aligned}$$

Chord f occurs in each of these three fundamental cut-sets, and there is no other fundamental cut-set that contains f. The converse of Theorem 4-5 is also true.

THEOREM 4-6

With respect to a given spanning tree T, a branch b_i that determines a fundamental cut-set S is contained in every fundamental circuit associated with the chords in S, and in no others.

Proof: The proof consists of arguments similar to those that led to Theorem 4-5. Let the fundamental cut-set S determined by a branch b_i be

$$S = \{b_i, c_1, c_2, \ldots, c_p\},$$

and let Γ_1 be the fundamental circuit determined by chord c_1:

$$\Gamma_1 = \{c_1, b_1, b_2, \ldots, b_q\}.$$

Since the number of edges common to S and Γ_1 must be even, b_i must be in Γ_1. The same is true for the fundamental circuits made by chords $c_2, c_3, \ldots, c_p$.

On the other hand, suppose that b_i occurs in a fundamental circuit Γ_{p+1} made by a chord other than $c_1, c_2, \ldots, c_p$. Since none of the chords $c_1, c_2, \ldots, c_p$ is in Γ_{p+1}, there is only one edge b_i common to a circuit Γ_{p+1} and the cut-set S, which is not possible. Hence the theorem. ■

Turning again for illustration to the graph in Fig. 4-3, consider branch e of spanning tree $\{b, c, e, h, k\}$. The fundamental cut-set determined by e is

$$\{e, d, f\}.$$

The two fundamental circuits determined by chords d and f are

$$\text{determined by chord } d: \quad \{d, c, e\},$$
$$\text{determined by chord } f: \quad \{f, e, h, k\}.$$

Branch e is contained in both these fundamental circuits, and none of the remaining three fundamental circuits contains branch e.

CONNECTIVITY AND SEPARABILITY

Edge Connectivity: Each cut-set of a connected graph G consists of a certain number of edges. The number of edges in the smallest cut-set (i.e., cut-set with fewest number of edges) is defined as the *edge connectivity* of G. Equivalently, the edge connectivity of a connected graph† can be defined as

the minimum number of edges whose removal (i.e., deletion) reduces the rank of the graph by one. The edge connectivity of a tree, for instance, is one. The edge connectivities of the graphs in Figs. 4-1(a), 4-3, 4-5 are one, two, and three, respectively.

Vertex Connectivity: On examining the graph in Fig. 4-5, we find that although removal of no single edge (or even a pair of edges) disconnects the

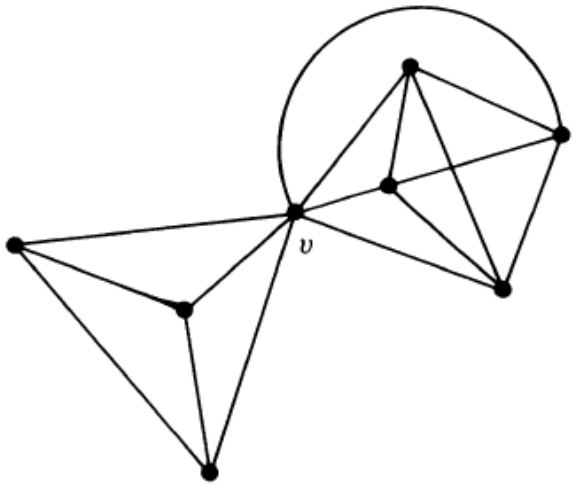


Fig. 4-5 Separable graph.

graph, the removal of the single vertex v does.† Therefore, we define another analogous term called *vertex connectivity*. The *vertex connectivity* (or simply *connectivity*) of a connected graph G is defined as the minimum number of vertices whose removal from G leaves the remaining graph disconnected.‡ Again, the vertex connectivity of a tree is one. The vertex connectivities of the graphs in Figs. 4-1(a), 4-3, and 4-5 are one, two, and one, respectively. Note that from the way we have defined it vertex connectivity is meaningful only for graphs that have three or more vertices and are not complete.

Separable Graph: A connected graph is said to be *separable* if its vertex connectivity is one. All other connected graphs are called *nonseparable*. An equivalent definition is that a connected graph G is said to be separable if there exists a subgraph g in G such that $\bar{g}$ (the complement of g in G) and g have only one vertex in common. That these two definitions are equivalent can be easily seen (Problem 4-7). In a separable graph a vertex whose removal disconnects the graph is called a *cut-vertex*, a *cut-node*, or an *articulation point*.

For example, in Fig. 4-5 the vertex v is a cut-vertex, and in Fig. 4-1(a) vertex v_4 is a cut-vertex. It can be shown (Problem 4-18) that in a tree every vertex with degree greater than one is a cut-vertex. Moreover:

THEOREM 4-7

A vertex v in a connected graph G is a cut-vertex if and only if there exist two vertices x and y in G such that every path between x and y passes through v.

The proof of the theorem is quite easy and is left as an exercise (Problem 4-17). The implication of the theorem is very significant. It states that v is a crucial vertex in the sense that any communication between x and y (if G represented a communication network) must "pass through" v.

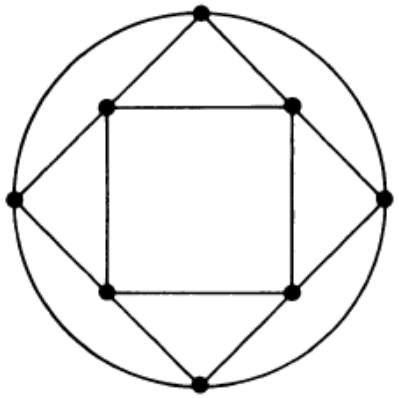


Fig. 4-6 Graph with 8 vertices and 16 edges.

An Application: Suppose we are given n stations that are to be connected by means of e lines (telephone lines, bridges, railroads, tunnels, or highways) where $e \geq n - 1$. What is the best way of connecting? By "best" we mean that the network should be as invulnerable to destruction of individual stations and individual lines as possible. In other words, construct a graph with n vertices and e edges that has the maximum possible edge connectivity and vertex connectivity.

For example, the graph in Fig. 4-5 has $n = 8$, $e = 16$, and has vertex connectivity of one and edge connectivity of three. Another graph with the same number of vertices and edges (8 and 16, respectively) can be drawn as shown in Fig. 4-6.

It can easily be seen that the edge connectivity as well as the vertex connectivity of this graph is four. Consequently, even after any three stations are bombed, or any three lines destroyed, the remaining stations can still con-

tinue to "communicate" with each other. Thus the network of Fig. 4-6 is better connected than that of Fig. 4-5 (although both consist of the same number of lines—16).

THEOREM 4-8

The edge connectivity of a graph G cannot exceed the degree of the vertex with the smallest degree in G.

Proof: Let vertex v_i be the vertex with the smallest degree in G. Let $d(v_i)$ be the degree of v_i. Vertex v_i can be separated from G by removing the $d(v_i)$ edges incident on vertex v_i. Hence the theorem. ■

THEOREM 4-9

The vertex connectivity of any graph G can never exceed the edge connectivity of G.

Proof: Let α denote the edge connectivity of G. Therefore, there exists a cut-set S in G with α edges. Let S partition the vertices of G into subsets V_1 and V_2. By removing at most α vertices from V_1 (or V_2) on which the edges in S are incident, we can effect the removal of S (together with all other edges incident on these vertices) from G. Hence the theorem. ■

COROLLARY

Every cut-set in a nonseparable graph with more than two vertices contains at least two edges.

THEOREM 4-10

The maximum vertex connectivity one can achieve with a graph G of n vertices and e edges ($e \geq n - 1$) is the integral part of the number $2e/n$; that is, $\lfloor 2e/n \rfloor$.

Proof: Every edge in G contributes two degrees. The total ($2e$ degrees) is divided among n vertices. Therefore, there must be at least one vertex in G whose degree is equal to or less than the number $2e/n$. The vertex connectivity of G cannot exceed this number, in light of Theorems 4-8 and 4-9.

To show that this value can actually be achieved, one can first construct an n-vertex regular graph of degree $\lfloor 2e/n \rfloor$ and then add the remaining $e - (n/2) \cdot \lfloor 2e/n \rfloor$ edges arbitrarily. The completion of the proof is left as an exercise. ■

The results of Theorems 4-8, 4-9, and 4-10 can be summarized as follows:

$$\text{vertex connectivity} \leq \text{edge connectivity} \leq \frac{2e}{n},$$

and

$$\text{maximum vertex connectivity possible} = \left\lfloor \frac{2e}{n} \right\rfloor.$$

Thus, for a graph with 8 vertices and 16 edges (Figs. 4-5 and 4-6), for example, we can achieve a vertex connectivity (and therefore edge connectivity) as high as four ($= 2 \cdot 16/8$).

A graph G is said to be *k-connected* if the vertex connectivity of G is k; therefore, a *1-connected* graph is the same as a separable graph.

THEOREM 4-11

A connected graph G is k-connected if and only if every pair of vertices in G is joined by k or more paths that do not intersect,† and at least one pair of vertices is joined by exactly k nonintersecting paths.

THEOREM 4-12

The edge connectivity of a graph G is k if and only if every pair of vertices in G is joined by k or more edge-disjoint paths (i.e., paths that may intersect, but have no edges in common), and at least one pair of vertices is joined by exactly k edge-disjoint paths.

The reader is referred to Chapter 5 of [1-5] for the proofs of Theorems 4-11 and 4-12. Note that our definition of k-connectedness is slightly different from the one given in [1-5]. A special result of Theorem 4-11 is that a graph G is nonseparable if and only if any pair of vertices in G can be placed in a circuit (Problem 4-13).

The reader is encouraged to verify these theorems by enumerating all edge-disjoint and vertex-disjoint paths between each of the 15 pairs of vertices in Fig. 4-3.

NETWORK FLOWS

In a network of telephone lines, highways, railroads, pipelines of oil (or gas or water), and so on, it is important to know the maximum rate of flow that is possible from one station to another in the network. This type of network is represented by a weighted connected graph in which the vertices are the stations and the edges are lines through which the given commodity (oil, gas, water, number of messages, number of cars, etc.) flows. The weight,

a real positive number, associated with each edge represents the capacity of the line, that is, the maximum amount of flow possible per unit of time. The graph in Fig. 4-7, for example, represents a flow network consisting of 12 stations and 31 lines. The capacity of each of these lines is also indicated in the figure.

It is assumed that at each intermediate vertex the total rate of commodity entering is equal to the rate leaving. In other words, there is no accumulation or generation of the commodity at any vertex along the way. Furthermore, the flow through a vertex is limited only by the capacities of the edges incident on it. In other words, the vertex itself can handle as much flow as allowed through the edges. Finally, the lines are lossless.

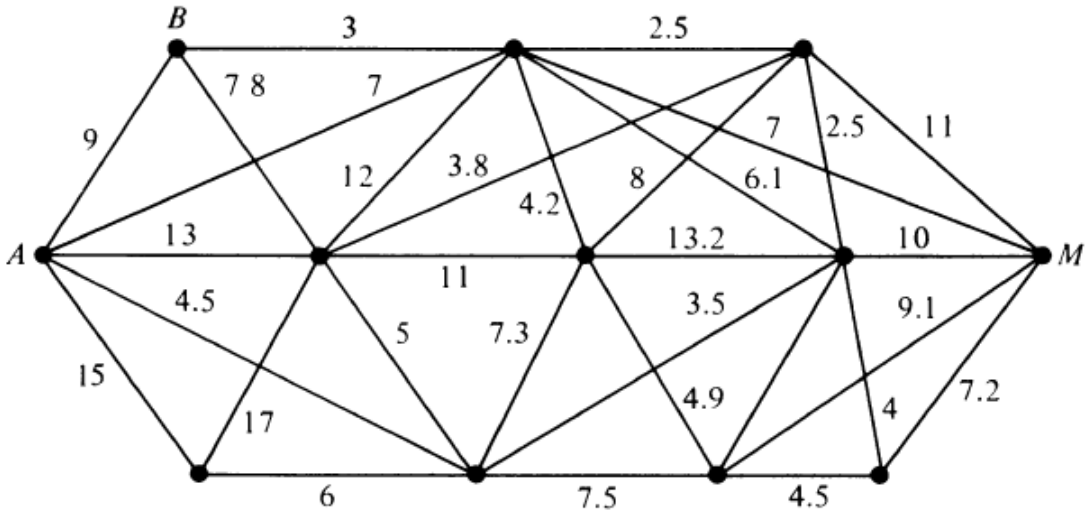


Fig. 4-7 Graph of a flow network.

In such a flow problem the questions to be answered are

1. What is the maximum flow possible through the network between a specified pair of vertices—say, from B to M in Fig. 4-7?

2. How do we achieve this flow (i.e., determine the actual flow through each edge when the maximum flow exists)?

Theorem 4-13, perhaps the most important result in the theory of transport networks, answers the first question. The second question is answered implicitly by a constructive proof of the theorem. To facilitate the statement and proof of the theorem, let us define a few terms.

A *cut-set with respect to a pair of vertices* a and b in a connected graph G puts a and b into two different components (i.e., separates vertices a and b). For instance, in Fig. 4-3 cut-set $\{d, e, f\}$ is a cut-set with respect to v_1 and v_6. The set $\{f, g, h\}$ is also a cut-set with respect to v_1 and v_6. But the cut-set $\{f, g, h\}$ is *not* a cut-set with respect to v_1 and v_6. The *capacity of cut-set* S in a weighted connected graph G (in which the weight of each edge represents its flow capacity) is defined as the sum of the weights of all the edges in S.

THEOREM 4-13

The maximum flow possible between two vertices a and b in a network is equal to the minimum of the capacities of all cut-sets with respect to a and b.

Proof: Consider any cut-set S with respect to vertices a and b in G. In the subgraph $G - S$ (the subgraph left after removing S from G) there is no path between a and b. Therefore, every path in G between a and b must contain at least one edge of S. Thus every flow from a to b (or from b to a) must pass through one or more edges of S. Hence the total flow rate between these two vertices cannot exceed the capacity of S. Since this holds for all cut-sets with respect to a and b, the flow rate cannot exceed the minimum of their capacities. ■

1-ISOMORPHISM

A separable graph consists of two or more nonseparable subgraphs. Each of the largest nonseparable subgraphs is called a *block*. (Some authors use the term *component*, but to avoid confusion with components of a disconnected graph, we shall use the term block.) The graph in Fig. 4-5 has two blocks. The graph in Fig. 4-8 has five blocks (and three cut-vertices a, b, and c); each block

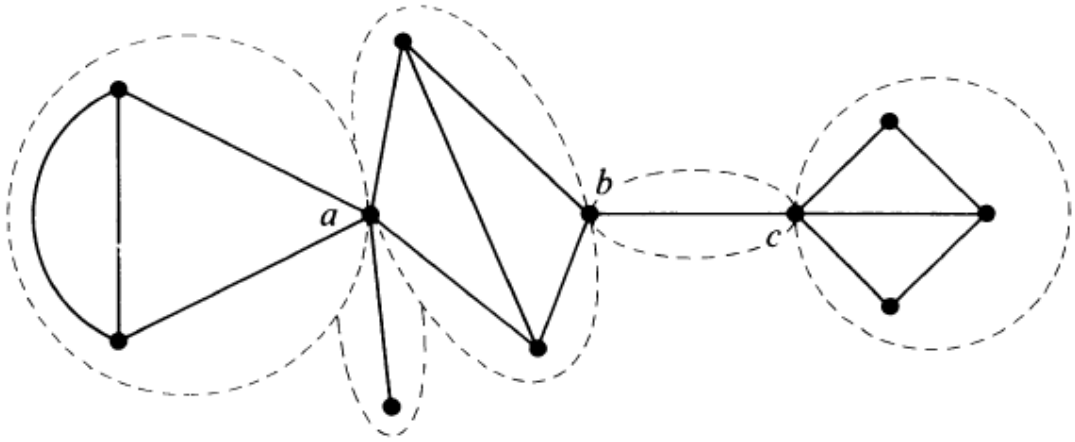


Fig. 4-8 Separable graph with three cut-vertices and five blocks.

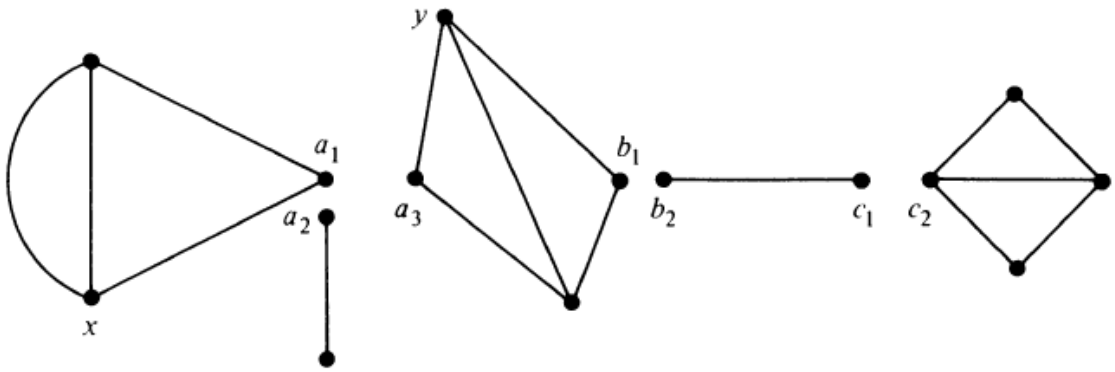


Fig. 4-9 Disconnected graph 1-isomorphic to Fig. 4-8.

is shown enclosed by a broken line. Note that a nonseparable connected graph consists of just one block.

Visually compare the disconnected graph in Fig. 4-9 with the one in Fig. 4-8. These two graphs are certainly not isomorphic (they do not have the same number of vertices), but they are related by the fact that the blocks of the graph in Fig. 4-8 are isomorphic to the components of the graph in Fig. 4-9. Such graphs are said to be *1-isomorphic*. More formally:

Two graphs G_1 and G_2 are said to be *1-isomorphic* if they become isomorphic to each other under repeated application of the following operation.

Operation 1: "Split" a cut-vertex into two vertices to produce two disjoint subgraphs.

From this definition it is apparent that two nonseparable graphs are 1-isomorphic if and only if they are isomorphic.

THEOREM 4-14

If G_1 and G_2 are two 1-isomorphic graphs, the rank of G_1 equals the rank of G_2 and the nullity of G_1 equals the nullity of G_2.

Proof: Under operation 1, whenever a cut-vertex in a graph G is "split" into two vertices, the number of components in G increases by one. Therefore, the rank of G which is

$$\text{number of vertices in } G - \text{number of components in } G$$

remains invariant under operation 1.

Also, since no edges are destroyed or new edges created by operation 1, two 1-isomorphic graphs have the same number of edges. Two graphs with equal rank and with equal numbers of edges must have the same nullity, because

$$\text{nullity} = \text{number of edges} - \text{rank}. \quad \blacksquare$$

What if we join two components of Fig. 4-9 by "gluing" together two vertices (say vertex x to y)? We obtain the graph shown in Fig. 4-10.

Clearly, the graph in Fig. 4-10 is 1-isomorphic to the graph in Fig. 4-9. Since the blocks of the graph in Fig. 4-10 are isomorphic to the blocks of the graph in Fig. 4-8, these two graphs are also 1-isomorphic. Thus the three graphs in Figs. 4-8, 4-9, and 4-10 are 1-isomorphic to one another.

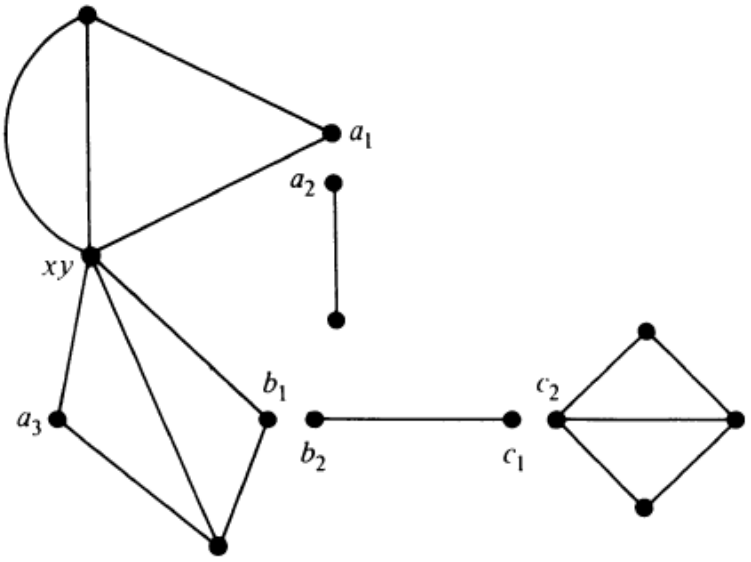


Fig. 4-10 Graph 1-isomorphic to Figs. 4-8 and 4-9.

2-ISOMORPHISM

In Section 4-7 we generalized the concept of isomorphism by introducing 1-isomorphism. A graph G_1 was 1-isomorphic to graph G_2 if the blocks of G_1 were isomorphic to the blocks of G_2. Since a nonseparable graph is just one block, 1-isomorphism for nonseparable graphs is the same as isomorphism. However, for separable graphs (i.e., graphs with vertex connectivity of one), 1-isomorphism is different from isomorphism. Graphs that are isomorphic are also 1-isomorphic, but 1-isomorphic graphs may not be isomorphic. This generalized isomorphism is very useful in the study of separable graphs.

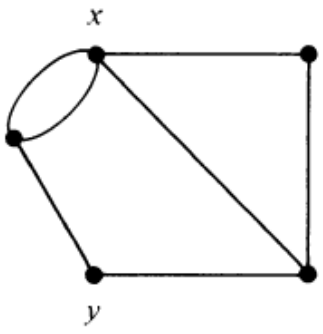
We can generalize this concept further to broaden its scope for 2-connected graphs (i.e., graphs with vertex connectivity of two), as follows:

In a 2-connected graph G let vertices x and y be a pair of vertices whose removal from G will leave the remaining graph disconnected. In other words, G consists of a subgraph g_1 and its complement $\bar{g}_1$ such that g_1 and $\bar{g}_1$ have exactly two vertices, x and y, in common. Suppose that we perform the following *operation 2* on G (after which, of course, G no longer remains the original graph).

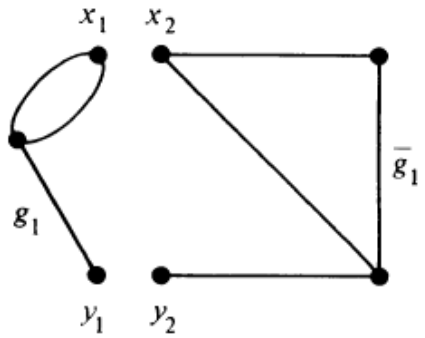
Operation 2: "Split" the vertex x into x_1 and x_2 and the vertex y into y_1 and y_2 such that G is split into g_1 and $\bar{g}_1$. Let vertices x_1 and y_1 go with g_1 and x_2 and y_2 with $\bar{g}_1$. Now rejoin the graphs g_1 and $\bar{g}_1$ by merging x_1 with y_2 and x_2 with y_1. (Clearly, edges whose end vertices were x and y in G could have gone with g_1 or $\bar{g}_1$, without affecting the final graph.)

Two graphs are said to be *2-isomorphic* if they become isomorphic after undergoing operation 1 (in Section 4-7) or operation 2, or both operations any number of times. For example, Fig. 4-11 shows how the two graphs in Figs. 4-11(a) and (d) are 2-isomorphic. Note that in (a) the degree of vertex x is four, but in (d) no vertex is of degree four.

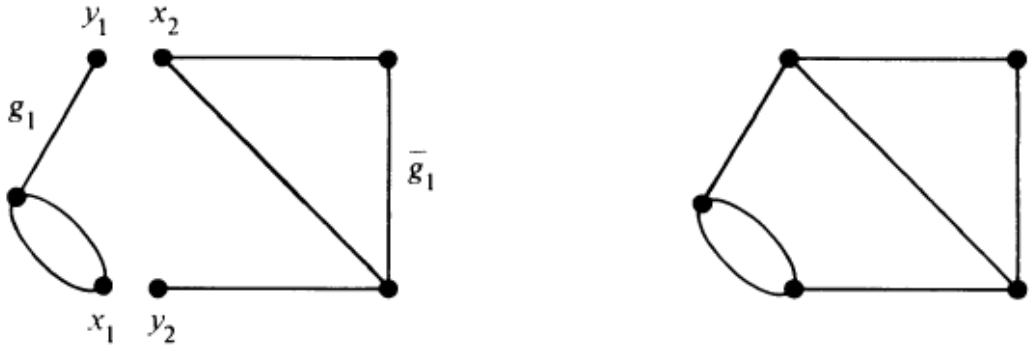
From the definition it follows immediately that isomorphic graphs are always 1-isomorphic, and 1-isomorphic graphs are always 2-isomorphic. But 2-isomorphic graphs are not necessarily 1-isomorphic, and 1-isomorphic



(a)



(b)



(c) (d)

Fig. 4-11 2-isomorphic graphs (a) and (d).

graphs are not necessarily isomorphic. However, for graphs with connectivity three or more, isomorphism, 1-isomorphism, and 2-isomorphism are synonymous.

It is clear that no edges or vertices are created or destroyed under operation 2. Therefore, the rank and nullity of a graph remain unchanged under operation 2. And as shown in Section 4-7, the rank or nullity of a graph does not change under operation 1. Therefore, 2-isomorphic graphs are equal in rank and equal in nullity. The fact that the rank r and nullity μ are not enough to specify a graph within 2-isomorphism can easily be shown by constructing a counterexample (Problem 4-23).

Circuit Correspondence: Two graphs G_1 and G_2 are said to have a *circuit correspondence* if they meet the following condition: There is a one-to-one correspondence between the edges of G_1 and G_2 and a one-to-one correspondence between the circuits of G_1 and G_2, such that a circuit in G_1 formed by certain edges of G_1 has a corresponding circuit in G_2 formed by the corresponding edges of G_2, and vice versa. Isomorphic graphs, obviously, have circuit correspondence.

Since in a separable graph G every circuit is confined to a particular block (Problem 4-15), every circuit in G retains its edges as G undergoes *operation 1* (in Section 4-7). Hence 1-isomorphic graphs have circuit correspondence.

Similarly, let us consider what happens to a circuit in a graph G when it undergoes operation 2, as defined in this section. A circuit Γ in G will fall in one of three categories:

1. Γ is made of edges all in g_1, or
2. Γ is made of edges all in $\bar{g}_1$, or
3. Γ is made of edges from both g_1 and $\bar{g}_1$, and in that case Γ must include both vertices x and y.

In cases 1 and 2, Γ is unaffected by operation 2. In case 3, Γ still has the original edges, except that the path between vertices x and y in g_1, which constituted a part of Γ, is "flipped around." Thus every circuit in a graph undergoing operation 2 retains its original edges. Therefore, 2-isomorphic graphs also have circuit correspondence.

Theorem 4-15, which is considered the most important result for 2-isomorphic graphs, is due to H. Whitney.

THEOREM 4-15

Two graphs are 2-isomorphic if and only if they have circuit correspondence.

Proof: The "only if" part has already been shown in the argument preceding the theorem. The "if" part is more involved, and the reader is referred to Whitney's original paper [4-7].

As we shall observe in subsequent chapters, the ideas of 2-isomorphism and circuit correspondence play important roles in the theory of contact networks, electrical networks, and in duality of graphs.

PLANAR GRAPHS

A graph G is said to be *planar* if there exists some geometric representation of G which can be drawn on a plane such that no two of its edges intersect.† A graph that cannot be drawn on a plane without a crossover between its edges is called *nonplanar*.

A drawing of a geometric representation of a graph on any surface such that no edges intersect is called *embedding*. Thus, to declare that a graph G is nonplanar, we have to show that of all possible geometric representations of G none can be embedded in a plane. Equivalently, a geometric graph G is planar if there exists a graph isomorphic to G that is embedded in a plane.

Otherwise, G is nonplanar. An embedding of a planar graph G on a plane is called a *plane representation* of G.

For instance, consider the graph represented by Fig. 1-3. The geometric representation shown in Fig. 1-3 clearly is not embedded in a plane, because the edges e and f are intersecting. But if we redraw edge f outside the quadrilateral, leaving the other edges unchanged, we have embedded the new geometric graph in the plane, thus showing that the graph which is being represented by Fig. 1-3 is planar. As another example, the two isomorphic diagrams in Fig. 2-2 are different geometric representations of one and the same graph. One of the diagrams is a plane representation; the other one is not. The graph, of course, is planar. On the other hand, you will not be able to draw any of the three configurations in Fig. 2-3 on a plane without edges intersecting. The reason is that the graph which these three different diagrams in Fig. 2-3 represent is nonplanar.

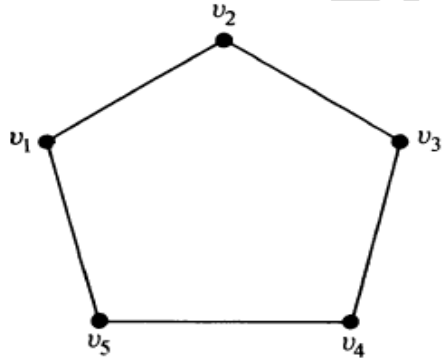
A natural question now is: How can we tell if a graph G [which may be given by an abstract notation $G = (V, E, \Psi)$ or by one of its geometric representations] is planar or nonplanar? To answer this question, let us first discuss two specific nonplanar graphs which are of fundamental importance. These are called Kuratowski's graphs, after the Polish mathematician Kasimir Kuratowski, who discovered their unique property.

KURATOWSKI'S TWO GRAPHS

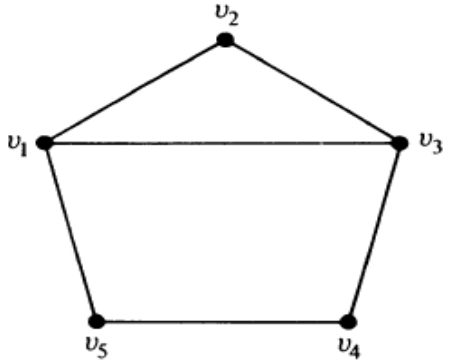
THEOREM 5-1

The complete graph of five vertices is nonplanar.

Proof: Let the five vertices in the complete graph be named v_1, v_2, v_3, v_4, and v_5. A complete graph, as you may recall, is a simple graph in which every vertex is joined to every other vertex by means of an edge. This being the case, we must



(a)



(b)

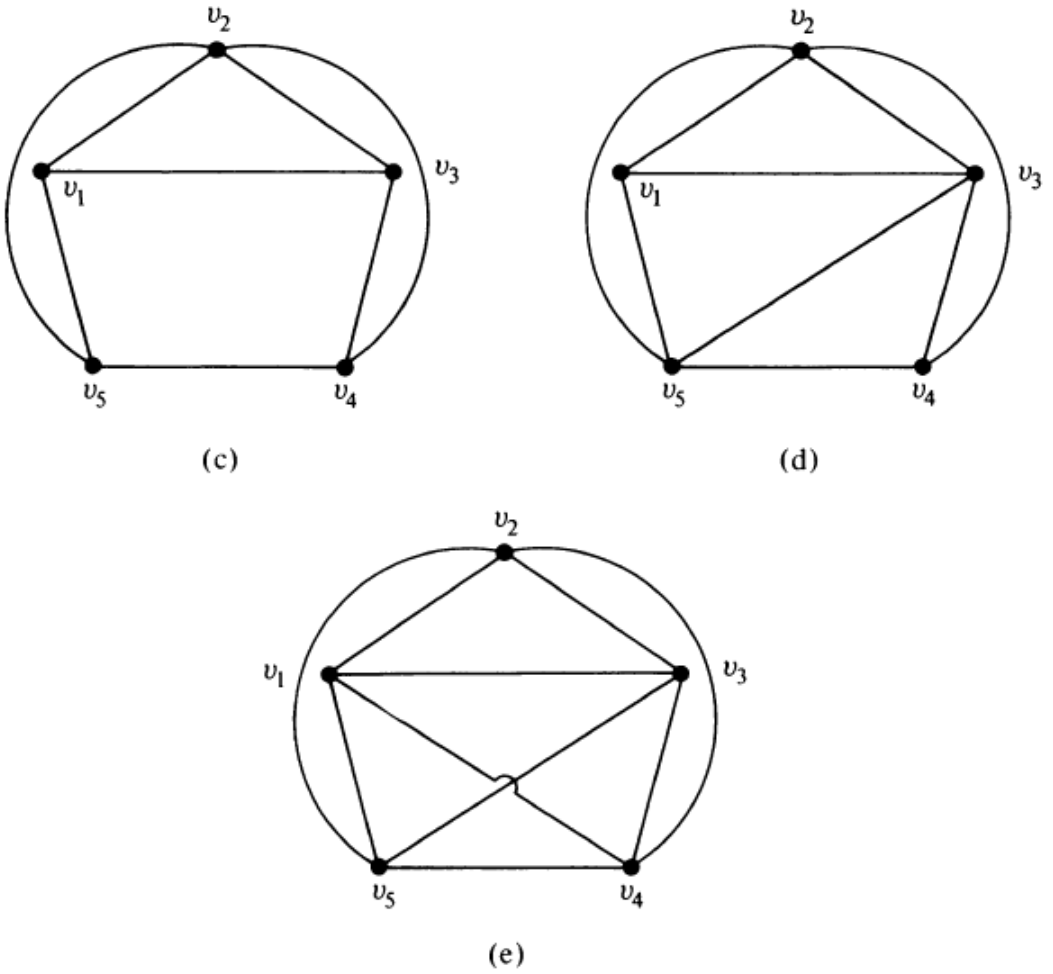


Fig. 5-1 Building up of the five-vertex complete graph.

have a circuit going from v_1 to v_2 to v_3 to v_4 to v_5 to v_1—that is, a pentagon. See Fig. 5-1(a). This pentagon must divide the plane of the paper into two regions, one *inside* and the other *outside* (Jordan curve theorem).

Since vertex v_1 is to be connected to v_3 by means of an edge, this edge may be drawn inside or outside the pentagon (without intersecting the five edges drawn previously). Suppose that we choose to draw a line from v_1 to v_3 inside the pentagon. See Fig. 5-1(b). (If we choose outside, we end up with the same argument.)

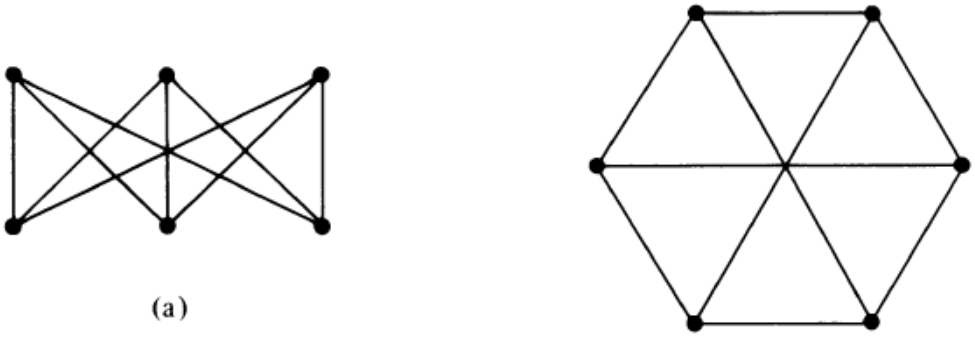
Now we have to draw an edge from v_2 to v_4 and another one from v_2 to v_5. Since neither of these edges can be drawn inside the pentagon without crossing over the edge already drawn, we draw both these edges outside the pentagon. See Fig. 5-1(c). The edge connecting v_3 and v_5 cannot be drawn outside the pentagon without crossing the edge between v_2 and v_4. Therefore, v_3 and v_5 have to be connected with an edge inside the pentagon. See Fig. 5-1(d).

Now we have yet to draw an edge between v_1 and v_4. This edge cannot be placed inside or outside the pentagon without a crossover. Thus the graph cannot be embedded in a plane. See Fig. 5-1(e). ■

Some readers may find this proof somewhat unsatisfactory because it depends so heavily on visual intuition. Do not despair; we shall provide you with an algebraic nonvisual proof in the next section.

A complete graph with five vertices is the first of the two graphs of Kuratowski. The second graph of Kuratowski is a regular† connected graph with six vertices and nine edges, shown in its two common geometric representations in Figs. 5-2(a) and (b), where it is fairly easy to see that the graphs are isomorphic.

Employing visual geometric arguments similar to those used in proving Theorem 5-1, it can be shown that the second graph of Kuratowski is also nonplanar. The proof of Theorem 5-2 is, therefore, left as an exercise (Problem 5-1).



(a) (b)

Fig. 5-2 Kuratowski's second graph.

THEOREM 5-2

Kuratowski's second graph is also nonplanar.

You may have noticed several properties common to the two graphs of Kuratowski. These are

1. Both are regular graphs.

2. Both are nonplanar.
3. Removal of one edge or a vertex makes each a planar graph.
4. Kuratowski's first graph is the nonplanar graph with the smallest number of vertices, and Kuratowski's second graph is the nonplanar graph with the smallest number of edges. Thus both are the simplest nonplanar graphs.

In the literature, Kuratowski's first graph is usually denoted by K_5 and the second graph by $K_{3,3}$—letter K being for Kuratowski.

DIFFERENT REPRESENTATIONS OF A PLANAR GRAPH

THEOREM 5-3

Any simple planar graph can be embedded in a plane such that every edge is drawn as a straight line segment.

Proof: The proof is involved and does not contribute much to the understanding of planarity. The interested reader is, therefore, referred to pages 74–77 in [1-2] or to the original paper of Fary [5-4]. As an illustration, the graph in Fig. 5-1(d) can be redrawn using straight line segments to look like Fig. 5-3. In this theorem, it is necessary for the graph to be simple because a self-loop or one of two parallel edges cannot be drawn by a straight line segment. ■

Region: A plane representation of a graph divides the plane into *regions* (also called *windows*, *faces*, or *meshes*), as shown in Fig. 5-4. A region is

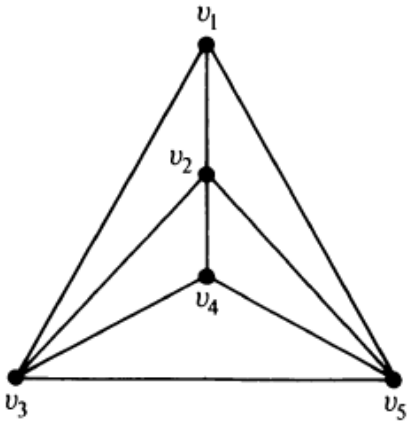


Fig. 5-3 Straight-line representation of the graph in Fig. 5-1(d).

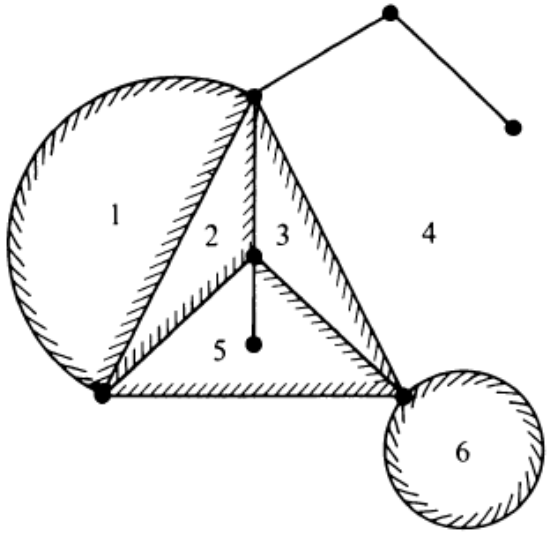


Fig. 5-4 Plane representation (the numbers stand for regions).

characterized by the set of edges (or the set of vertices) forming its *boundary*. Note that a region is not defined in a nonplanar graph or even in a planar graph not embedded in a plane. For example, the geometric graph in Fig. 1-3 does not have regions. Thus a region is a property of the specific plane representation of a graph and not of an abstract graph per se.

Infinite Region: The portion of the plane lying outside a graph embedded in a plane, such as region 4 in Fig. 5-4, is infinite in its extent. Such a region is called the *infinite*, *unbounded*, *outer*, or *exterior* region for that particular plane representation. Like other regions, the infinite region is also characterized by a set of edges (or vertices). Clearly, by changing the embedding of a given planar graph, we can change the infinite region. For instance, Figs. 5-1(d) and 5-3 are two different embeddings of the same graph. The finite region $v_1\, v_3\, v_5$ in Fig. 5-1(d) becomes the infinite region in Fig. 5-3. In fact, we shall shortly show that any region can be made the infinite region by proper embedding.

Embedding on a Sphere: To eliminate the distinction between finite and infinite regions, a planar graph is often embedded in the surface of a sphere. It is accomplished by stereographic projection of a sphere on a plane. Put the sphere on the plane and call the point of contact SP (south pole). At point SP, draw a straight line perpendicular to the plane, and let the point where this line intersects the surface of the sphere be called NP (north pole). See Fig. 5-5.

Now, corresponding to any point p on the plane, there exists a unique point p' on the sphere and vice versa, where p' is the point at which the straight line from point p to point NP intersects the surface of the sphere. Thus there is a one-to-one correspondence between the points of the sphere and the finite points on the plane, and points at infinity in the plane correspond to the point NP on the sphere.

From this construction, it is clear that any graph that can be embedded in

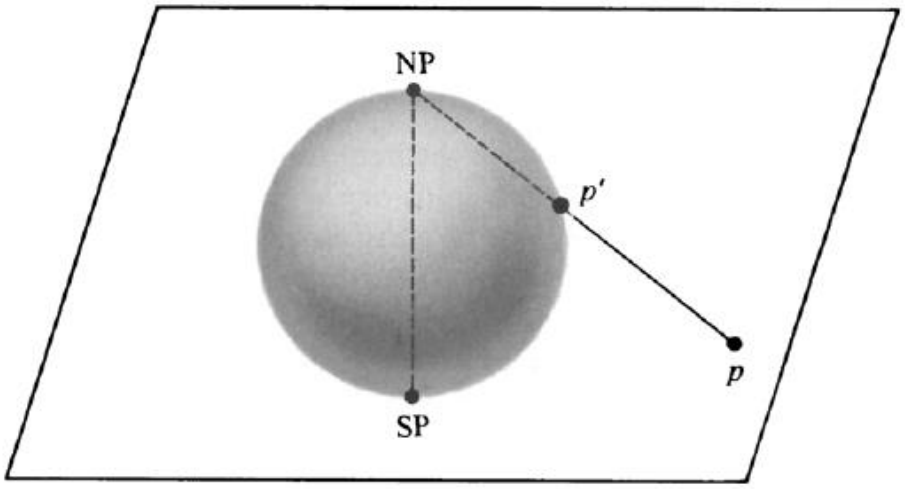


Fig. 5-5 Stereographic projection.

a plane (i.e., drawn on a plane such that its edges do not intersect) can also be embedded in the surface of the sphere, and vice versa. Hence

THEOREM 5-4

A graph can be embedded in the surface of a sphere if and only if it can be embedded in a plane.

A planar graph embedded in the surface of a sphere divides the surface into different regions. Each region on the sphere is finite, the infinite region on the plane having been mapped onto the region containing the point NP. Now it is clear that by suitably rotating the sphere we can make any specified region map onto the infinite region on the plane. From this we obtain

THEOREM 5-5

A planar graph may be embedded in a plane such that any specified region (i.e., specified by the edges forming it) can be made the infinite region.

Thinking in terms of the regions on the sphere, we see that there is no real difference between the infinite region and the finite regions on the plane. Therefore, when we talk of the regions in a plane regresentation of a graph, we include the infinite region. Also, since there is no essential difference between an embedding of a planar graph on a plane or on a sphere (a plane may be regarded as the surface of a sphere of infinitely large radius), the term "plane representation" of a graph is often used to include spherical as well as planar embedding.

Euler's Formula: Since a planar graph may have different plane representations, we may ask if the number of regions resulting from each embedding is the same. The answer is *yes*. Theorem 5-6, known as Euler's formula, gives the number of regions in any planar graph.

THEOREM 5-6

A connected planar graph with n vertices and e edges has $e - n + 2$ regions.

Proof: It will suffice to prove the theorem for a simple graph, because adding a self-loop or a parallel edge simply adds one region to the graph and simultaneously increases the value of e by one. We can also disregard (i.e., remove) all edges that do not form boundaries of any region. Three such edges are shown in Fig. 5-4. Addition (or removal) of any such edge increases (or decreases) e by one and increases (or decreases) n by one, keeping the quantity $e - n$ unaltered.

Since any simple planar graph can have a plane representation such that each edge is a straight line (Theorem 5-3), any planar graph can be drawn such that each region is a polygon (a polygonal net). Let the polygonal net representing the given graph consist of f regions or faces, and let k_p be the number of p-sided regions. Since each edge is on the boundary of exactly two regions,

$$3 \cdot k_3 + 4 \cdot k_4 + 5 \cdot k_5 + \cdots + r \cdot k_r = 2 \cdot e, \tag{5-1}$$

where k_r is the number of polygons, with maximum edges.
Also,

$$k_3 + k_4 + k_5 + \cdots + k_r = f. \tag{5-2}$$

The sum of all angles subtended at each vertex in the polygonal net is

$$2\pi n. \tag{5-3}$$

Recalling that the sum of all interior angles of a p-sided polygon is $\pi(p-2)$, and the sum of the exterior angles is $\pi(p+2)$, let us compute the expression in (5-3) as the grand sum of all interior angles of $f-1$ finite regions plus the sum of the exterior angles of the polygon defining the infinite region. This sum is

$$\begin{aligned}&\pi(3-2)\cdot k_3 + \pi(4-2)\cdot k_4 + \cdots + \pi(r-2)\cdot k_r + 4\pi \\ &= \pi(2e-2f) + 4\pi. \end{aligned} \tag{5-4}$$

Equating (5-4) to (5-3), we get

$$2\pi(e-f) + 4\pi = 2\pi n,$$

or

$$e - f + 2 = n.$$

Therefore, the number of regions is

$$f = e - n + 2. \quad \blacksquare$$

COROLLARY

In any simple, connected planar graph with f regions, n vertices, and e edges ($e > 2$), the following inequalities must hold:

$$e \geq \frac{3}{2}f, \tag{5-5}$$

$$e \leq 3n - 6. \tag{5-6}$$

Proof: Since each region is bounded by at least three edges and each edge belongs to exactly two regions,

$$2e \geq 3f$$

or

$$e \geq \frac{3}{2}f.$$

Substituting for f from Euler's formula in inequality (5-5),

$$e \geq \frac{3}{2}(e - n + 2)$$

or

$$e \leq 3n - 6. \quad \blacksquare$$

Inequality (5-6) is often useful in finding out if a graph is nonplanar. For example, in the case of K_5, the complete graph of five vertices [Fig. 5-1(e)],

$$n = 5, \qquad e = 10, \qquad 3n - 6 = 9 < e.$$

Thus the graph violates inequality (5-6), and hence it is not planar.

Incidentally, this is an alternative and independent proof of the nonplanarity of Kuratowski's first graph, as promised in Section 5-3.

The reader must be warned that inequality (5-6) is only a necessary, but *not* a sufficient, condition for the planarity of a graph. In other words, although every simple planar graph must satisfy (5-6), the mere satisfaction of this inequality does not guarantee the planarity of a graph. For example, Kuratowski's second graph, $K_{3,3}$, satisfies (5-6), because

$$e = 9,$$
$$3n - 6 = 3 \cdot 6 - 6 = 12.$$

Yet the graph is nonplanar.

To prove the nonplanarity of Kuratowski's second graph, we make use of the additional fact that no region in this graph can be bounded with fewer than four edges. Hence, if this graph were planar, we would have

$$2e \geq 4f,$$

and, substituting for f from Euler's formula,

$$2e \geq 4(e - n + 2),$$
or $$2 \cdot 9 \geq 4(9 - 6 + 2),$$
or $$18 \geq 20, \quad \text{a contradiction.}$$

Hence the graph cannot be planar.

Plane Representation and Connectivity: In a disconnected graph the embedding of each component can be considered independently. Therefore, it is clear that a disconnected graph is planar if and only if each of its components is planar. Similarly, in a separable (or 1-connected) graph the embedding of each block (i.e., maximal nonseparable subgraph) can be considered independently. Hence a separable graph is planar if and only if each of its blocks is planar.

Therefore, in questions of embedding or planarity, one need consider only nonseparable graphs.

THEOREM 5-7

The spherical embedding of every planar 3-connected graph is unique.

This theorem plays a very important role in determining if a graph is

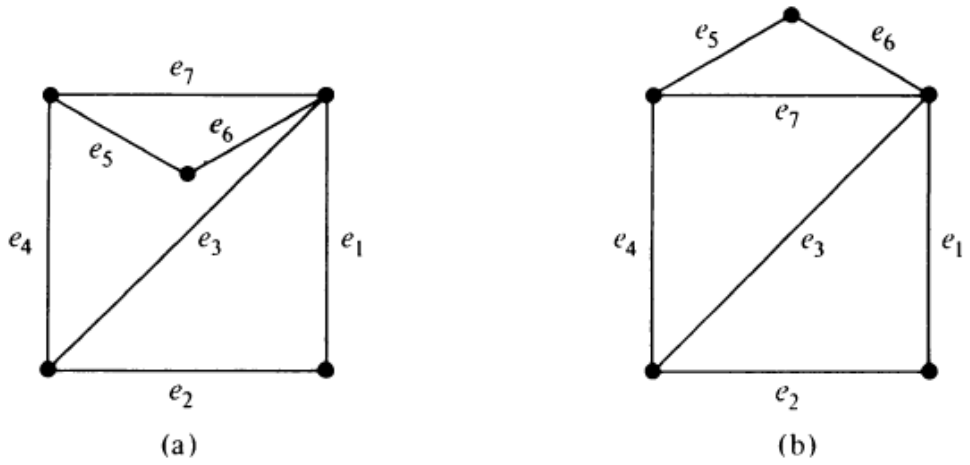


Fig. 5-6 Two distinct plane representations of the same graph.

planar or not. The theorem states that a 3-connected graph, if it can be embedded at all, can be embedded in only one way.

DETECTION OF PLANARITY

How to tell if a given graph G is planar or nonplanar is an important problem, and "find out by drawing it" is obviously not a good answer. We must have some simple and efficient criterion. Toward that goal, we take the following simplifying steps:

Elementary Reduction

Step 1: Since a disconnected graph is planar if and only if each of its components is planar, we need consider only one component at a time. Also, a separable graph is planar if and only if each of its blocks is planar. Therefore, for the given arbitrary graph G, determine the set

$$G = \{G_1, G_2, \ldots, G_k\},$$

where each G_i is a nonseparable block of G. Then we have to test each G_i for planarity.

Step 2: Since addition or removal of self-loops does not affect planarity,

remove all self-loops.

Step 3: Since parallel edges also do not affect planarity, eliminate edges in parallel by removing all but one edge between every pair of vertices.

Step 4: Elimination of a vertex of degree two by merging two edges in series† does not affect planarity. Therefore, eliminate all edges in series.

Repeated application of steps 3 and 4 will usually reduce a graph drastically. For example, Fig. 5-7 illustrates the series-parallel reduction of the graph of Fig. 5-6(b).

Let the nonseparable connected graph G_i be reduced to a new graph H_i after the repeated application of steps 3 and 4. What will graph H_i look like? Theorem 5-8 has the answer.

THEOREM 5-8

Graph H_i is

1. A single edge, or
2. A complete graph of four vertices, or
3. A nonseparable, simple graph with $n \geq 5$ and $e \geq 7$.

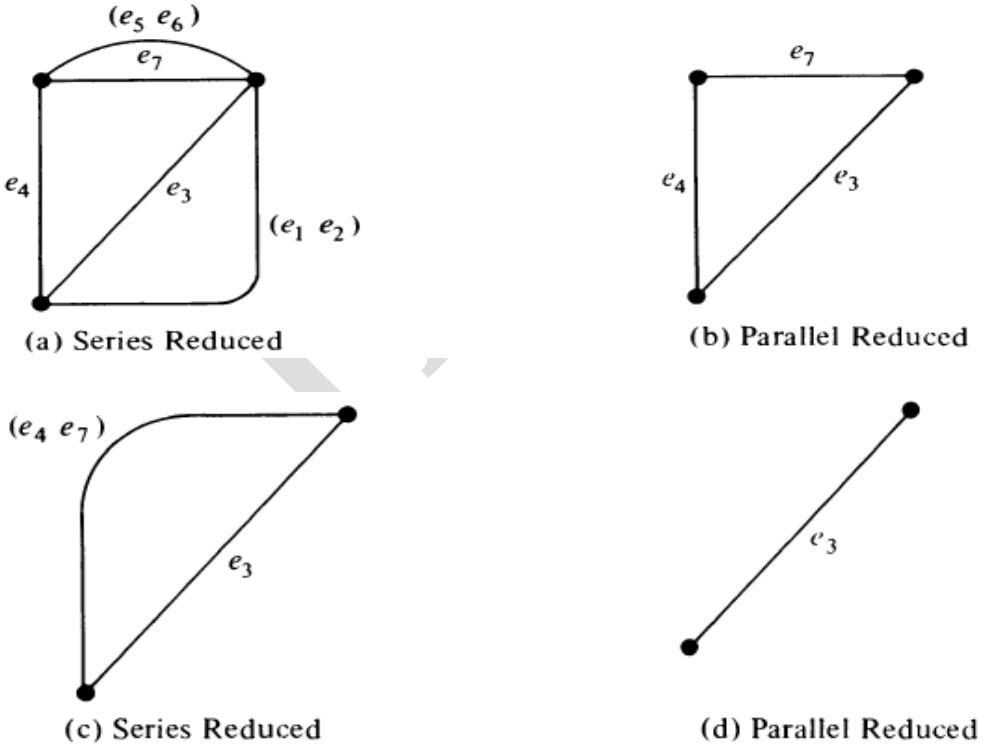


Fig. 5-7 Series-parallel reduction of the graph in Fig. 5-6(b).

Proof: The theorem can be proved by considering all connected nonseparable graphs of six edges or less. The proof is left as an exercise (Problem 5-9).

In Theorem 5-8, all H_i falling in categories 1 or 2 are planar and need not be checked further.

From now on, therefore, we need to investigate only *simple*, *connected*, *nonseparable graphs of at least five vertices and with every vertex of degree three or more*. Next, we can check to see if $e \leq 3n - 6$. If this inequality is not satisfied, the graph H_i is nonplanar. If the inequality is satisfied, we have to test the graph further and, with this, we come to Kuratowski's theorem (Theorem 5-9), perhaps the most important result of this chapter.

Homeomorphic Graphs: Two graphs are said to be *homeomorphic* if one graph can be obtained from the other by the creation of edges in series (i.e., by insertion of vertices of degree two) or by the merger of edges in series. The three graphs in Fig. 5-8 are homeomorphic to each other, for instance. A graph G is planar if and only if every graph that is homeomorphic to G is planar. (This is a restatement of series reduction, step 4 in this section.)

THEOREM 5-9

A necessary and sufficient condition for a graph G to be planar is that G does not contain either of Kuratowski's two graphs or any graph homeomorphic to either of them.

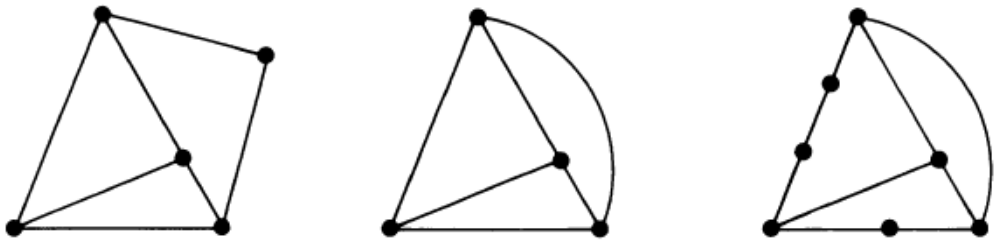


Fig. 5-8 Three graphs homeomorphic to each other.

Proof: The necessary condition is clear, because a graph G cannot be embedded in a plane if G has a subgraph that cannot be embedded. That this condition is also sufficient is surprising, and its proof is involved. Several different proofs of the theorem have appeared since Kuratowski stated and proved it in 1930. For a complete proof of the theorem, the reader is referred to Harary [1-5], pages 108–112, Berge [1-1], pages 211–213, or Busacker and Saaty [1-2], pages 70–73.

Note that it is *not* necessary for a nonplanar graph to have either of the Kuratowski graphs as a subgraph, as this theorem is sometimes misstated. The nonplanar graph may have a subgraph homeomorphic to a Kuratowski graph. For example, the graph in Fig. 5-9(a) is nonplanar, and yet it does not have either of the Kuratowski graphs as a subgraph. However, if we remove

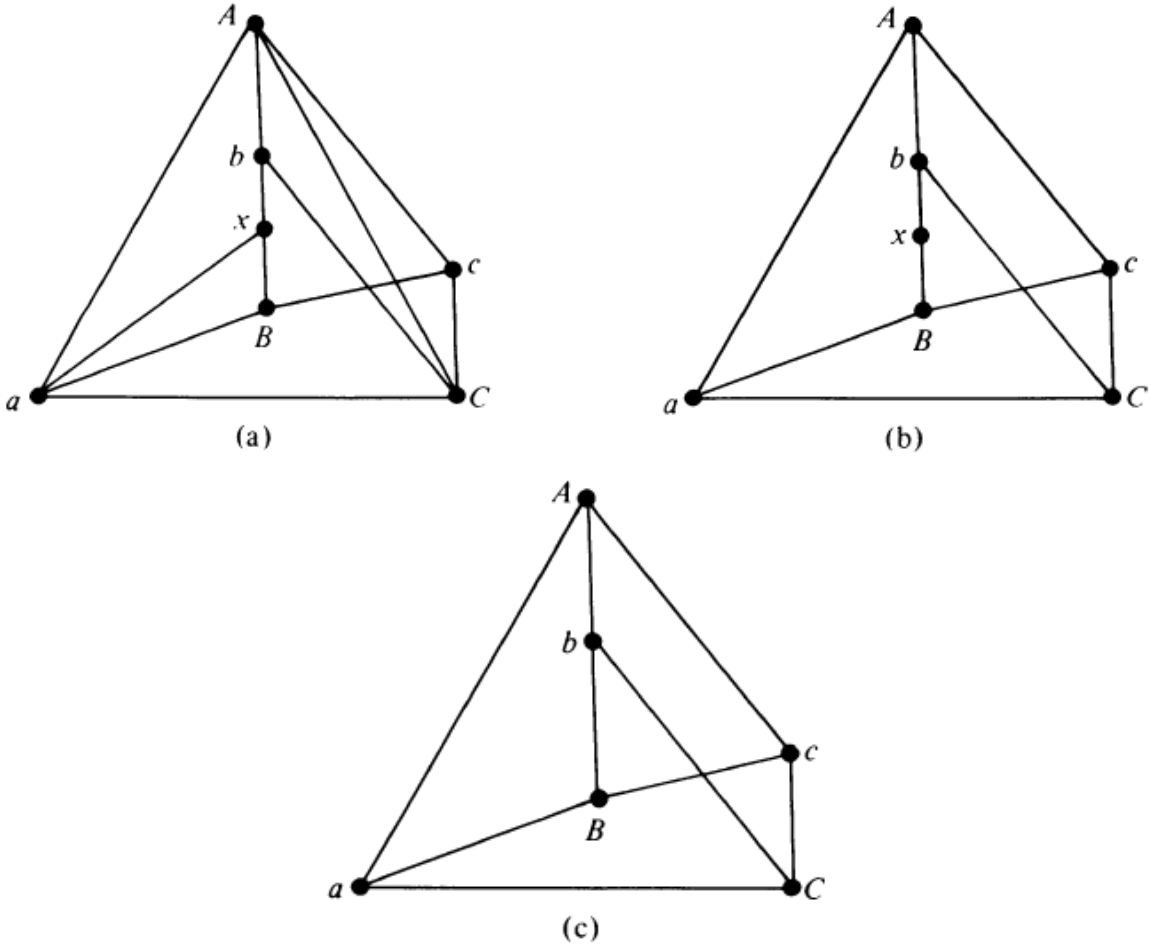


Fig. 5-9 Nonplanar graph with a subgraph homeomorphic to $K_{3,3}$.

edges (a, x) and (A, C) from this graph, we get a subgraph, as shown in Fig. 5-9(b). This subgraph is homeomorphic (merge two series edges at vertex x) to the one shown in Fig. 5-9(c). The graph of Fig. 5-9(c) clearly is isomorphic to $K_{3,3}$, Kuratowski's second graph, and this demonstrates the nonplanarity of the graph in Fig. 5-9(a).

Possible Questions

2 Mark Questions:

1. What is Fermat's Factorization method.
2. Prove that if p is a prime, then $a^p \equiv a(\text{mod } p)$ for any integer a.
3. Verify that $18^6 \equiv 1(\text{mod } 7^k)$ for $k = 1,2,3$.
4. Find the remainder when $15!$ is divided by 17.
5. Write about $\tau(n)$ and $\sigma(n)$ with example.
6. What is multiplicative function.
7. Prove that if f is a multiplicative function and F is defined by

$$F(n) = \sum_{d|n} f(d),$$

then F is also multiplicative.

8. Define Dirichlet Product.
9. Find the remainder when 5^{11} is divided by 7.
10. Use Fermat's method to factor 23449.

8 Mark Questions:

1. State and prove Fermat's Little theorem.
2. Prove that if p and q are distinct primes such that $a^p \equiv a(\text{mod } q)$ and $a^q \equiv a(\text{mod } p)$, then $a^{pq} \equiv a(\text{mod } pq)$.
3. State and prove Wilson's theorem.
4. Prove that the quadratic congruence $x^2 + 1 \equiv 0(\text{mod } p)$, where p is an odd prime, has a solution if and only if $p \equiv 1(\text{mod } 4)$.

5. Prove that if $n = p_1^{k_1} p_2^{k_2} \dots p_r^{k_r}$ is the prime factorization of $n > 1$, then the positive divisors of n are precisely those integers d of the form

$$d = p_1^{a_1} p_2^{a_2} \dots p_r^{a_r},$$

where $0 \le a_i \le k_i (i = 1,2,\dots,r)$.

6. Prove that if $n = p_1^{k_1} p_2^{k_2} \dots p_r^{k_r}$ is the prime factorization of $n > 1$, then

a) $\tau(n) = (k_1 + 1)(k_2 + 1)\dots(k_r + 1)$, and

b) $\sigma(n) = \dfrac{p_1^{k_1+1} - 1}{p_1 - 1} \dfrac{p_2^{k_2+1} - 1}{p_2 - 1} \dots \dfrac{p_r^{k_r+1} - 1}{p_r - 1}$.

7. Prove that the function τ and σ are both multiplicative functions.

8. Prove that if $\gcd(m,n) = 1$, then the set of positive divisors of mn consists of all products $d_1 d_2$, where $d_1 | n, d_2 | m$ and $\gcd(d_1, d_2) = 1$; furthermore, these products are all distinct.

9. Discuss about Dirichlet Product.

10. Find the remainder when 72^{1001} is divisible by 31.

11. Prove that the quadratic congruence $x^2 \equiv -1 (\text{mod } p)$, p is a prime, has a solution if and only if $p \equiv 1 (\text{mod } 4)$.

UNIT-III

SYLLABUS

Incidence matrix – Sub matrices – Circuit Matrix – Path Matrix – Adjacency Matrix – Chromatic Numb
Chromatic partitioning – Chromatic polynomial - Matching - Covering – Four Color Problem.

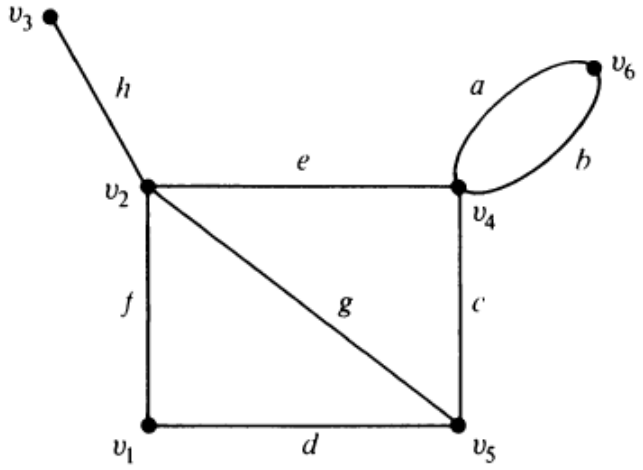
INCIDENCE MATRIX

Let G be a graph with n vertices, e edges, and no self-loops. Define an n by e matrix $\mathsf{A} = [a_{ij}]$, whose n rows correspond to the n vertices and the e columns correspond to the e edges, as follows:

The matrix element

$a_{ij} = 1$, if jth edge e_j is incident on ith vertex v_i, and

$= 0$, otherwise.



(a)

	a	b	c	d	e	f	g	h
v_1	0	0	0	1	0	1	0	0
v_2	0	0	0	0	1	1	1	1
v_3	0	0	0	0	0	0	0	1
v_4	1	1	1	0	1	0	0	0
v_5	0	0	1	1	0	0	1	0
v_6	1	1	0	0	0	0	0	0

(b)

Fig. 7-1 Graph and its incidence matrix.

Such a matrix A is called the *vertex-edge incidence matrix*, or simply *incidence matrix*. Matrix A for a graph G is sometimes also written as $A(G)$. A graph and its incidence matrix are shown in Fig. 7-1.

The incidence matrix contains only two elements, 0 and 1. Such a matrix is called a *binary matrix* or a *(0, 1)-matrix*. Let us stipulate that these two elements are from Galois field modulo 2.† Given any geometric representation of a graph without self-loops, we can readily write its incidence matrix.

On the other hand, if we are given an incidence matrix $A(G)$, we can construct its geometric graph G without ambiguity. The incidence matrix and the geometric graph contain the same information†—they are simply two alternative ways of representing the same (abstract) graph.

The following observations about the incidence matrix A can readily be made:

1. Since every edge is incident on exactly two vertices, each column of A has exactly two 1's.
2. The number of 1's in each row equals the degree of the corresponding vertex.
3. A row with all 0's, therefore, represents an isolated vertex.
4. Parallel edges in a graph produce identical columns in its incidence matrix, for example, columns 1 and 2 in Fig. 7-1.
5. If a graph G is disconnected and consists of two components g_1 and g_2, the incidence matrix $A(G)$ of graph G can be written in a block-diagonal form as

$$A(G) = \left[\begin{array}{c|c} A(g_1) & 0 \\ \hline 0 & A(g_2) \end{array}\right], \qquad (7\text{-}1)$$

where $A(g_1)$ and $A(g_2)$ are the incidence matrices of components g_1 and g_2. This observation results from the fact that no edge in g_1 is incident on vertices of g_2, and vice versa. Obviously, this remark is also true for a disconnected graph with any number of components.

6. Permutation of any two rows or columns in an incidence matrix simply corresponds to relabeling the vertices and edges of the same graph. This observation leads us to Theorem 7-1.

THEOREM 7-1

Two graphs G_1 and G_2 are isomorphic if and only if their incidence matrices $A(G_1)$ and $A(G_2)$ differ only by permutations of rows and columns.

Rank of the Incidence Matrix: Each row in an incidence matrix $A(G)$ may be regarded as a vector over GF(2) in the vector space of graph G. Let the vector in the first row be called A_1, in the second row A_2, and so on. Thus

$$A(G) = \begin{bmatrix} A_1 \\ A_2 \\ \cdot \\ \cdot \\ \cdot \\ A_n \end{bmatrix}, \qquad (7\text{-}2)$$

Since there are exactly two 1's in every column of A, the sum of all these vectors is 0 (this being a modulo 2 sum of the corresponding entries). Thus vectors $A_1, A_2, \ldots, A_n$ are not linearly independent. Therefore, the rank of A is less than n; that is, rank $A \leq n - 1$.

Now consider the sum of any m of these n vectors ($m \leq n - 1$). If the graph is connected, $A(G)$ cannot be partitioned, as in Eq. (7-1), such that $A(g_1)$ is with m rows and $A(g_2)$ with $n - m$ rows. In other words, no m by m submatrix of $A(G)$ can be found, for $m \leq n - 1$, such that the modulo 2 sum of those m rows is equal to zero.

Since there are only two constants 0 and 1 in this field, the additions of all vectors taken m at a time for $m = 1, 2, \ldots, n - 1$ exhausts all possible linear combinations of $n - 1$ row vectors. Thus we have just shown that no linear combination of m row vectors of A (for $m \leq n - 1$) can be equal to zero. Therefore, the rank of $A(G)$ must be at least $n - 1$.

Since the rank of $A(G)$ is no more than $n - 1$ and is no less than $n - 1$, it must be exactly equal to $n - 1$. Hence Theorem 7-2.

THEOREM 7-2

If $A(G)$ is an incidence matrix of a connected graph G with n vertices, the rank of $A(G)$ is $n - 1$.

The argument leading to Theorem 7-2 can be extended to prove that the rank of $A(G)$ is $n - k$, if G is a disconnected graph with n vertices and k components (Problem 7-3). This is the reason why the number $n - k$ has been called the rank of a graph with k components.

If we remove any one row from the incidence matrix of a connected graph, the remaining $(n - 1)$ by e submatrix is of rank $n - 1$ (Theorem 7-2). In other words, the remaining $n - 1$ row vectors are linearly independent. Thus we need only $n - 1$ rows of an incidence matrix to specify the corresponding graph completely, for $n - 1$ rows contain the same amount of information as the entire matrix. (This is obvious, since given $n - 1$ rows we can easily reconstitute the missing row, because each column in the matrix has exactly two 1's.)

Such an $(n - 1)$ by e submatrix A_f of A is called a *reduced incidence matrix*. The vertex corresponding to the deleted row in A_f is called the *reference vertex*. Clearly, any vertex of a connected graph can be made the reference vertex.

Since a tree is a connected graph with n vertices and $n - 1$ edges, its reduced incidence matrix is a square matrix of order and rank $n - 1$. In other words,

COROLLARY

The reduced incidence matrix of a tree is nonsingular.

A graph with n vertices and $n - 1$ edges that is not a tree is disconnected. The rank of the incidence matrix of such a graph will be less than $n - 1$. Therefore, the $(n - 1)$ by $(n - 1)$ reduced incidence matrix of such a graph will not be nonsingular. In other words, the reduced incidence matrix of a graph is nonsingular if and only if the graph is a tree.

SUBMATRICES OF A(G)

Let g be a subgraph of a graph G, and let $A(g)$ and $A(G)$ be the incidence matrices of g and G, respectively. Clearly, $A(g)$ is a submatrix of $A(G)$ (possibly with rows or columns permuted). In fact, there is a one-to-one correspondence between each n by k submatrix of $A(G)$ and a subgraph of G with k edges, k being any positive integer less than e and n being the number of vertices in G.

Submatrices of $A(G)$ corresponding to special types of subgraphs, such as circuits, spanning trees, or cut-sets in G, will undoubtedly exhibit special properties. Theorem 7-3 gives one such property.

THEOREM 7-3

Let $A(G)$ be an incidence matrix of a connected graph G with n vertices. An $(n-1)$ by $(n-1)$ submatrix of $A(G)$ is nonsingular if and only if the $n-1$ edges corresponding to the $n-1$ columns of this matrix constitute a spanning tree in G.

Proof: Every square submatrix of order $n-1$ in $A(G)$ is the reduced incidence matrix of the same subgraph in G with $n-1$ edges, and vice versa. From the remarks following Theorem 7-2, it is clear that a square submatrix of $A(G)$ is nonsingular if and only if the corresponding subgraph is a tree. The tree in this case is a spanning tree, because it contains $n-1$ edges of the n-vertex graph. Thus the theorem. ■

CIRCUIT MATRIX

Let the number of different circuits in a graph G be q and the number of edges in G be e. Then a *circuit matrix* $B = [b_{ij}]$ of G is a q by e, (0, 1)-matrix defined as follows:

$$b_{ij} = 1, \quad \text{if } i\text{th circuit includes } j\text{th edge, and}$$
$$= 0, \quad \text{otherwise.}$$

To emphasize the fact that B is a circuit matrix of graph G, the circuit matrix may also be written as $B(G)$.

The graph in Fig. 7-1(a) has four different circuits, $\{a, b\}$, $\{c, e, g\}$, $\{d, f, g\}$, and $\{c, d, f, e\}$. Therefore, its circuit matrix is a 4 by 8, (0, 1)-matrix as shown:

$$\mathsf{B}(G) = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} \end{array}. \tag{7-3}$$

The following observations can be made about a circuit matrix $\mathsf{B}(G)$ of a graph G:

1. A column of all zeros corresponds to a noncircuit edge (i.e., an edge that does not belong to any circuit).
2. Each row of $\mathsf{B}(G)$ is a circuit vector.
3. Unlike the incidence matrix, a circuit matrix is capable of representing a self-loop—the corresponding row will have a single 1.
4. The number of 1's in a row is equal to the number of edges in the corresponding circuit.
5. If graph G is separable (or disconnected) and consists of two blocks (or components) g_1 and g_2, the circuit matrix $\mathsf{B}(G)$ can be written in a block-diagonal form as

$$\mathsf{B}(G) = \left[\begin{array}{c|c} \mathsf{B}(g_1) & 0 \\ \hline 0 & \mathsf{B}(g_2) \end{array}\right],$$

where $\mathsf{B}(g_1)$ and $\mathsf{B}(g_2)$ are the circuit matrices of g_1 and g_2. This observation results from the fact that circuits in g_1 have no edges belonging to g_2, and vice versa (Problem 4-14).

6. Permutation of any two rows or columns in a circuit matrix simply corresponds to relabeling the circuits and edges.
7. Two graphs G_1 and G_2 will have the same circuit matrix if and only if G_1 and G_2 are 2-isomorphic (Theorem 4-15). In other words, (unlike

an incidence matrix) the circuit matrix does not specify a graph completely. It only specifies the graph within 2-isomorphism. For instance, it can be easily verified that the two graphs in Figs. 4-11(a) and (d) have the same circuit matrix, yet the graphs are not isomorphic.

An important theorem relating the incidence matrix and the circuit matrix of a self-loop-free graph G is

THEOREM 7-4

Let B and A be, respectively, the circuit matrix and the incidence matrix (of a self-loop-free graph) whose columns are arranged using the same order of edges. Then every row of B is orthogonal to every row A; that is,

$$\mathsf{A} \cdot \mathsf{B}^T = \mathsf{B} \cdot \mathsf{A}^T = 0 \pmod 2, \tag{7-4}$$

where superscript T denotes the transposed matrix.

Proof: Consider a vertex v and a circuit Γ in the graph G. Either v is in Γ or it is not. If v is not in Γ, there is no edge in the circuit Γ that is incident on v. On the other hand, if v is in Γ, the number of those edges in the circuit Γ that are incident on v is exactly two.

With this remark in mind, consider the ith row in A and the jth row in B. Since the edges are arranged in the same order, the nonzero entries in the corresponding positions occur only if the particular edge is incident on the ith vertex and is also in the jth circuit.

If the ith vertex is not in the jth circuit, there is no such nonzero entry, and the dot product of the two rows is zero. If the ith vertex is in the jth circuit, there will be exactly two 1's in the sum of the products of individual entries. Since $1 + 1 = 0 \pmod 2$, the dot product of the two arbitrary rows—one from A and the other from B—is zero. Hence the theorem. ■

As an example, let us multiply the incidence matrix and transposed circuit of the graph in Fig. 7-1(a), after making sure that the edges are in the same order in both.

$$A \cdot B^T = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

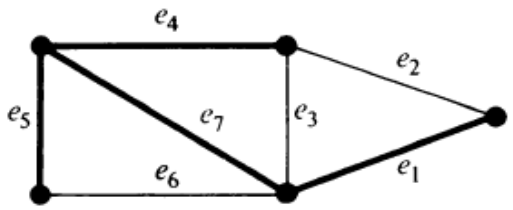
$$= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (\text{mod } 2).$$

FUNDAMENTAL CIRCUIT MATRIX AND RANK OF *B*

A submatrix (of a circuit matrix) in which all rows correspond to a set of fundamental circuits is called a *fundamental circuit matrix* B_f. A graph and its fundamental circuit matrix with respect to a spanning tree (indicated by heavy lines) are shown in Fig. 7-2.

As in matrices A and B, permutations of rows (and/or of columns) do not affect B_f. If n is the number of vertices and e the number of edges in a connected graph, then B_f is an $(e - n + 1)$ by e matrix, because the number of fundamental circuits is $e - n + 1$, each fundamental circuit being produced by one chord.

Let us arrange the columns in B_f such that all the $e - n + 1$ chords correspond to the first $e - n + 1$ columns. Furthermore, let us rearrange the rows such that the first row corresponds to the fundamental circuit made



(a)

e_2	e_3	e_6	e_1	e_4	e_5	e_7
1	0	0	1	1	0	1
0	1	0	0	1	0	1
0	0	1	0	0	1	1

(b)

Fig. 7-2 Graph and its fundamental circuit matrix (with respect to the spanning tree shown in heavy lines).

by the chord in the first column, the second row to the fundamental circuit made by the second, and so on. This indeed is how the fundamental circuit matrix is arranged in Fig. 7-2(b).

A matrix B_f thus arranged can be written as

$$\mathsf{B}_f = [\mathsf{I}_\mu \mid \mathsf{B}_t], \tag{7-5}$$

where I_μ is an identity matrix of order $\mu = e - n + 1$, and B_t is the remaining μ by $(n - 1)$ submatrix, corresponding to the branches of the spanning tree.

From Eq. (7-5) it is clear that the

$$\text{rank of } \mathsf{B}_f = \mu = e - n + 1.$$

Since B_f is a submatrix of the circuit matrix B, the

$$\text{rank of } \mathsf{B} \geq e - n + 1.$$

In fact, we can prove Theorem 7-5.

THEOREM 7-5

If B is a circuit matrix of a connected graph G with e edges and n vertices,

$$\text{rank of } \mathsf{B} = e - n + 1.$$

Proof: If A is an incidence matrix of G, from Eq. (7-4) we have

$$\mathsf{A} \cdot \mathsf{B}^T = 0 \pmod 2.$$

Therefore, according to Sylvester's theorem (Appendix B),

$$\text{rank of } \mathsf{A} + \text{rank of } \mathsf{B} \leq e;$$

that is,

$$\text{rank of } \mathsf{B} \leq e - \text{rank of } \mathsf{A}.$$

Since $\text{rank of } \mathsf{A} = n - 1$

we have $\text{rank of } \mathsf{B} \leq e - n + 1.$

But $\text{rank of } \mathsf{B} \geq e - n + 1.$

Therefore, we must have

$$\text{rank of } \mathsf{B} = e - n + 1. \blacksquare$$

An Alternative Proof: Theorem 7-5 can also be proved by considering the circuit subspace W_Γ in the vector space W_G of a graph, as discussed in Chapter 6.

Every row in circuit matrix B is a vector in W_Γ, and since the rank of any matrix is equal to the number of linearly independent rows (or columns) in the matrix, we have.

$$\text{rank of matrix } \mathsf{B} = \text{number of linearly independent rows in } \mathsf{B};$$

but the number of linearly independent rows in $\mathsf{B} \leq$ number of linearly independent vectors in W_Γ, and the number of linearly independent vectors in W_Γ = dimension of $W_\Gamma = \mu$. Therefore, rank of $\mathsf{B} \leq e - n + 1$. Since we already showed that rank of $\mathsf{B} \geq e - n + 1$, Theorem 7-5 follows. ■

Note that in talking of spanning trees of a graph G it is necessary to assume that G is connected. In the case of a disconnected graph, we would have to consider a spanning forest and fundamental circuits with respect to this forest. It is not difficult to show (considering component by component) that if G is a disconnected graph with k components, e edges, and n vertices,

$$\text{rank of } \mathsf{B} = \mu = e - n + k.$$

PATH MATRIX

Another (0, 1)-matrix often convenient to use in communication and transportation networks is the *path matrix*. A path matrix is defined for a specific pair of vertices in a graph, say (x, y), and is written as $\mathsf{P}(x, y)$. The rows in $\mathsf{P}(x, y)$ correspond to different paths between vertices x and y, and the columns correspond to the edges in G. That is, the path matrix for (x, y) vertices is $\mathsf{P}(x, y) = [p_{ij}]$, where

$$\begin{aligned} p_{ij} &= 1, \quad \text{if } j\text{th edge lies in } i\text{th path, and} \\ &= 0, \quad \text{otherwise.} \end{aligned}$$

As an illustration, consider all paths between vertices v_3 and v_4 in Fig. 7-1(a). There are three different paths; $\{h, e\}$, $\{h, g, c\}$, and $\{h, f, d, c\}$. Let us number them 1, 2, and 3, respectively. Then we get the 3 by 8 path matrix $\mathsf{P}(v_3, v_4)$:

$$\mathsf{P}(v_3, v_4) = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \end{array}.$$

Some of the observations one can make at once about a path matrix $\mathsf{P}(x, y)$ of a graph G are

1. A column of all 0's corresponds to an edge that does not lie in any path between x and y.
2. A column of all 1's corresponds to an edge that lies in every path between x and y.
3. There is no row with all 0's.
4. The ring sum of any two rows in $\mathsf{P}(x, y)$ corresponds to a circuit or an edge-disjoint union of circuits.

THEOREM 7-7

If the edges of a connected graph are arranged in the same order for the columns of the incidence matrix A and the path matrix $\mathsf{P}(x, y)$, then the product (mod 2)

$$\mathsf{A} \cdot \mathsf{P}^T(x, y) = \mathsf{M},$$

where the matrix M has 1's in two rows x and y, and the rest of the $n-2$ rows are all 0's.

Proof: The proof is left as an exercise for the reader (Problem 7-14).

As an example, multiply the incidence matrix in Fig. 7-1 to the transposed $\mathsf{P}(v_3, v_4)$, just discussed.

$$\mathsf{A}\cdot\mathsf{P}^T(v_3, v_4) = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$= \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{array} \pmod 2.$$

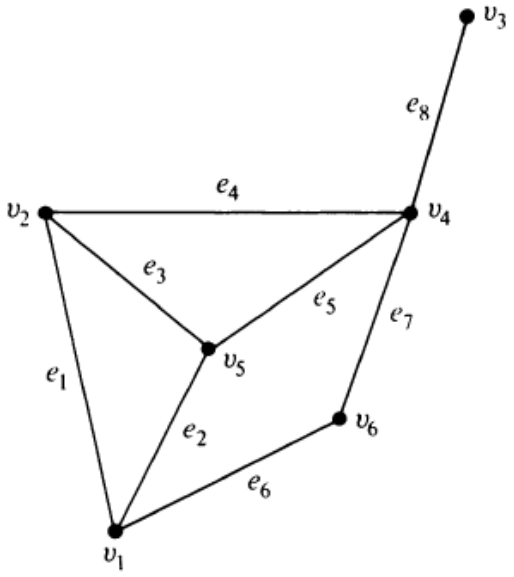
Other properties of the path matrix, such as the rank, are left for the reader to investigate on his own. It should be noted that a path matrix contains less information about the graph in general than any of the matrices A, B, or C does.

ADJACENCY MATRIX

As an alternative to the incidence matrix, it is sometimes more convenient to represent a graph by its *adjacency matrix* or *connection matrix*. The adjacency matrix of a graph G with n vertices and no parallel edges is an n by n symmetric binary matrix $\mathsf{X} = [x_{ij}]$ defined over the ring of integers such that

$x_{ij} = 1$, if there is an edge between ith and jth vertices, and

$= 0$, if there is no edge between them.



$$X = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

Fig. 7-7 Simple graph and its adjacency matrix.

A simple graph and its adjacency matrix are shown in Fig. 7-7.

Observations that can be made immediately about the adjacency matrix X of a graph G are

1. The entries along the principal diagonal of X are all 0's if and only if the graph has no self-loops. A self-loop at the ith vertex corresponds to $x_{ii} = 1$.
2. The definition of adjacency matrix makes no provision for parallel edges. This is why the adjacency matrix X was defined for graphs without parallel edges.†

3. If the graph has no self-loops (and no parallel edges, of course), the degree of a vertex equals the number of 1's in the corresponding row or column of X.
4. Permutations of rows and of the corresponding columns imply reordering the vertices. It must be noted, however, that the rows and columns must be arranged in the same order. Thus, if two rows are interchanged in X, the corresponding columns must also be interchanged. Hence two graphs G_1 and G_2 with no parallel edges are isomorphic if and only if their adjacency matrices $\mathsf{X}(G_1)$ and $\mathsf{X}(G_2)$ are related:

$$\mathsf{X}(G_2) = \mathsf{R}^{-1} \cdot \mathsf{X}(G_1) \cdot \mathsf{R},$$

where R is a permutation matrix.

5. A graph G is disconnected and is in two components g_1 and g_2 if and only if its adjacency matrix $\mathsf{X}(G)$ can be partitioned as

$$\mathsf{X}(G) = \begin{bmatrix} \mathsf{X}(g_1) & 0 \\ 0 & \mathsf{X}(g_2) \end{bmatrix},$$

where $\mathsf{X}(g_1)$ is the adjacency matrix of the component g_1 and $\mathsf{X}(g_2)$ is that of the component g_2.

This partitioning clearly implies that there exists no edge joining any vertex in subgraph g_1 to any vertex in subgraph g_2.

6. Given any square, symmetric, binary matrix Q of order n, one can always construct a graph G of n vertices (and no parallel edges) such that Q is the adjacency matrix of G.

Powers of X: Let us multiply by itself the 6 by 6 adjacency matrix of the simple graph in Fig. 7-7. The result, another 6 by 6 symmetric matrix X^2, is shown below (note that this is ordinary matrix multiplication in the ring of integers and *not* mod 2 multiplication):

$$X^2 = \begin{bmatrix} 3 & 1 & 0 & 3 & 1 & 0 \\ 1 & 3 & 1 & 1 & 2 & 2 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 3 & 1 & 0 & 4 & 1 & 0 \\ 1 & 2 & 1 & 1 & 3 & 2 \\ 0 & 2 & 1 & 0 & 2 & 2 \end{bmatrix}.$$

The value of an off-diagonal entry in X^2, that is, ijth entry ($i \neq j$) in X^2,
= number of 1's in the dot product of ith row and jth column (or jth row) of X.
= number of positions in which both ith and jth rows of X have 1's.
= number of vertices that are adjacent to both ith and jth vertices.
= number of different paths of length two between ith and jth vertices.

Similarly, the ith diagonal entry in X^2 is the number of 1's in the ith row (or column) of matrix X. Thus the value of each diagonal entry in X^2 equals the degree of the corresponding vertex, if the graph has no self-loops.

Since a matrix commutes with matrices that are its own power,

$$X \cdot X^2 = X^2 \cdot X = X^3.$$

And since the product of two square symmetric matrices that commute is also a symmetric matrix, X^3 is a symmetric matrix. (Again note that this is an ordinary product and not mod 2.)

The matrix X^3 for the graph of Fig. 7-7 is

$$X^3 = \begin{bmatrix} 2 & 7 & 3 & 2 & 7 & 6 \\ 7 & 4 & 1 & 8 & 5 & 2 \\ 3 & 1 & 0 & 4 & 1 & 0 \\ 2 & 8 & 4 & 2 & 8 & 7 \\ 7 & 5 & 1 & 8 & 4 & 2 \\ 6 & 2 & 0 & 7 & 2 & 0 \end{bmatrix}.$$

Let us now consider the ijth entry of X^3.

ijth entry of $\mathbf{X}^3$ = dot product of ith row $\mathbf{X}^2$ and jth column (or row) of $\mathbf{X}$.

$$= \sum_{k=1}^{n} ik\text{th entry of } \mathbf{X}^2 \cdot kj\text{th entry of } \mathbf{X}.$$

$$= \sum_{k=1}^{n} \text{number of all different edge sequences}^{\dagger} \text{ of three edges from } i\text{th to } j\text{th vertex via } k\text{th vertex.}$$

= number of different edge sequences of three edges between ith and jth vertices.

For example, consider how the 1,5th entry on $\mathbf{X}^3$ for the graph of Fig. 7-7 is formed. It is given by the dot product

$$\text{row 1 of } \mathbf{X}^2 \cdot \text{row 5 of } \mathbf{X} = (3, 1, 0, 3, 1, 0) \cdot (1, 1, 0, 1, 0, 0)$$
$$= 3 + 1 + 0 + 3 + 0 + 0 = 7.$$

These seven different edge sequences of three edges between v_1 and v_5 are

$$\{e_1, e_1, e_2\}, \quad \{e_2, e_2, e_2\}, \quad \{e_6, e_6, e_2\}, \quad \{e_2, e_3, e_3\},$$
$$\{e_6, e_7, e_5\}, \quad \{e_2, e_5, e_5\}, \quad \{e_1, e_4, e_5\}.$$

Clearly this list includes all the paths of length three between v_1 and v_5, that is, $\{e_6, e_7, e_5\}$ and $\{e_1, e_4, e_5\}$.

It is left as an exercise for the reader to show (Problem 7-19) that the iith entry in $\mathbf{X}^3$ equals twice the number of different circuits of length three (i.e., triangles) in the graph passing through the corresponding vertex v_i.

The general result that includes the properties of $\mathbf{X}$, $\mathbf{X}^2$, and $\mathbf{X}^3$ discussed so far is expressed in Theorem 7-8.

THEOREM 7-8

Let $\mathbf{X}$ be the adjacency matrix of a simple graph G. Then the ijth entry in $\mathbf{X}^r$ is the number of different edge sequences of r edges between vertices v_i and v_j.

Proof: The theorem holds for $r = 1$, and it has been proved for $r = 2$ and 3 also. It can be proved for any positive integer r, by induction.

In other words, assume that it holds for $r - 1$, and then evaluate the ijth entry in $\mathbf{X}$, with the help of the relation

$$X^r = X^{r-1} \cdot X,$$

as was done for X^3.

COROLLARY A

In a connected graph, the distance between two vertices v_i and v_j (for $i \neq j$) is k, if and only if k is the smallest integer for which the i,jth entry in x^k is nonzero.

This is a useful result in determining the distances between different pairs of vertices.

COROLLARY B

If X is the adjacency matrix of a graph G with n vertices, and

$$Y = X + X^2 + X^3 + \cdots + X^{n-1}, \qquad \text{(in the ring of integers),}$$

then G is disconnected if and only if there exists at least one entry in matrix Y that is zero.

Relationship Between $A(G)$ *and* $X(G)$*:* Recall that if a graph G has no self-loops, its incidence matrix $A(G)$ contains all the information about G. Likewise, if G has no parallel edges, its adjacency matrix $X(G)$ contains all the information about G. Therefore, if a graph G has neither self-loops nor parallel edges (i.e., G is a simple graph), both $A(G)$ and $X(G)$ contain the entire information. Thus it is natural to expect that either matrix can be obtained directly from the other, in the case of a simple graph. This relationship is given in Problem 7-23.

CHROMATIC NUMBER

Painting all the vertices of a graph with colors such that no two adjacent vertices have the same color is called the *proper coloring* (or sometimes simply *coloring*) of a graph. A graph in which every vertex has been assigned a color

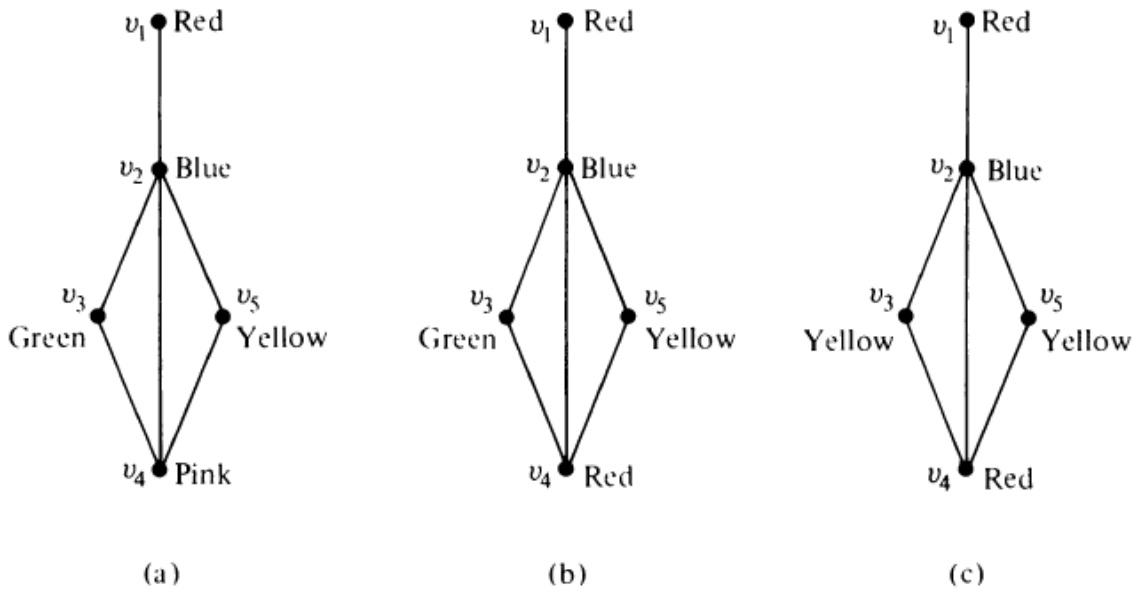


Fig. 8-1 Proper colorings of a graph.

according to a proper coloring is called a *properly colored* graph. Usually a given graph can be properly colored in many different ways. Figure 8-1 shows three different proper colorings of a graph.

The proper coloring which is of interest to us is one that requires the minimum number of colors. A graph G that requires κ different colors for its proper coloring, and no less, is called a κ-*chromatic* graph, and the number κ is called the *chromatic number* of G. You can verify that the graph in Fig. 8-1 is 3-chromatic.

In coloring graphs there is no point in considering disconnected graphs. How we color vertices in one component of a disconnected graph has no effect on the coloring of the other components. Therefore, it is usual to investigate coloring of connected graphs only. All parallel edges between two vertices can be replaced by a single edge without affecting adjacency of vertices. Self-loops must be disregarded. Thus for coloring problems we need to consider only simple, connected graphs.

Some observations that follow directly from the definitions just introduced are

1. A graph consisting of only isolated vertices is 1-chromatic.
2. A graph with one or more edges (not a self-loop, of course) is at least *2-chromatic* (also called *bichromatic*).

3. A complete graph of n vertices is n-chromatic, as all its vertices are adjacent. Hence a graph containing a complete graph of r vertices is at least r-chromatic. For instance, every graph having a triangle is at least 3-chromatic.

4. A graph consisting of simply one circuit with $n \geq 3$ vertices is 2-chromatic if n is even and 3-chromatic if n is odd. (This can be seen by numbering vertices 1, 2, . . . , n in sequence and assigning one color to odd vertices and another to even. If n is even, no adjacent vertices will have the same color. If n is odd, the nth and first vertex will be adjacent and will have the same color, thus requiring a third color for proper coloring.)

Proper coloring of a given graph is simple enough, but a proper coloring with the minimum number of colors is, in general, a difficult task. In fact, there has not yet been found a simple way of characterizing a κ-chromatic graph. (The brute-force method of using all possible combinations can, of course, always be applied, as in any combinatorial problem. But brute force is highly unsatisfactory, because it gets out of hand as soon as the size of the graph increases beyond a few vertices.) Chromatic numbers of some specific types of graphs will be discussed in the rest of this section.

THEOREM 8-1

Every tree with two or more vertices is 2-chromatic.

Proof: Select any vertex v in the given tree T. Consider T as a rooted tree at vertex v. Paint v with color 1. Paint all vertices adjacent to v with color 2. Next, paint the vertices adjacent to these (those that just have been colored with 2) using color 1. Continue this process till every vertex in T has been painted. (See Fig. 8-2). Now in T we find that all vertices at odd distances from v have color 2, while v and vertices at even distances from v have color 1.

Now along any path in T the vertices are of alternating colors. Since there is one and only one path between any two vertices in a tree, no two adjacent vertices have the same color. Thus T has been properly colored with two colors. One color would not have been enough (observation 2 in this section). ■

Though a tree is 2-chromatic, not every 2-chromatic graph is a tree. (The utilities graph, for instance, is not a tree.) What then is the characterization

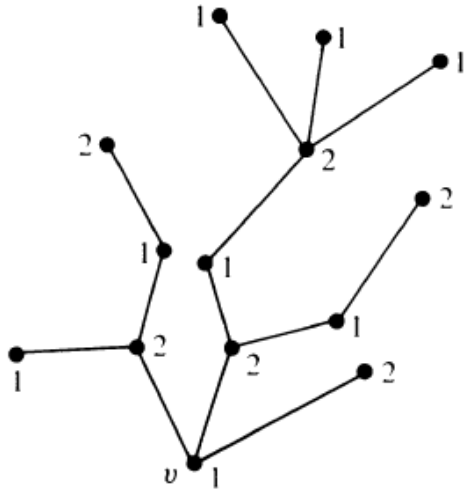


Fig. 8-2 Proper coloring of a tree.

of a 2-chromatic graph? Theorem 8-2 (due to König) characterizes all 2-chromatic graphs.

THEOREM 8-2

A graph with at least one edge is 2-chromatic if and only if it has no circuits of odd length.

Proof: Let G be a connected graph with circuits of only even lengths. Consider a spanning tree T in G. Using the coloring procedure and the result of Theorem 8-1, let us properly color T with two colors. Now add the chords to T one by one. Since G had no circuits of odd length, the end vertices of every chord being replaced are differently colored in T. Thus G is colored with two colors, with no adjacent vertices having the same color. That is, G is 2-chromatic.

Conversely, if G has a circuit of odd length, we would need at least three colors just for that circuit (observation 4 in this section). Thus the theorem. ■

An upper limit on the chromatic number of a graph is given by Theorem 8-3, whose proof is left as an exercise (Problem 8-1).

THEOREM 8-3

If $d_{\max}$ is the maximum degree of the vertices in a graph G,

$$\text{chromatic number of } G \leq 1 + d_{\max}.$$

Brooks [8-1] showed that this upper bound can be improved by 1 if G has no complete graph of $d_{\max} + 1$ vertices. In that case

chromatic number of $G \leq d_{\max}$.

A graph G is called *bipartite* if its vertex set V can be decomposed into two disjoint subsets V_1 and V_2 such that every edge in G joins a vertex in V_1 with a vertex in V_2. Thus every tree is a bipartite graph. So are the graphs in Figs. 8-6 and 8-8. Obviously, a bipartite graph can have no self-loop. A set of parallel edges between a pair of vertices can all be replaced with one edge without affecting bipartiteness of a graph.

Clearly, every 2-chromatic graph is bipartite because the coloring partitions the vertex set into two subsets V_1 and V_2 such that no two vertices in V_1 (or V_2) are adjacent. Similarly, every bipartite graph is 2-chromatic, with one trivial exception; a graph of two or more isolated vertices and with no edges is bipartite but is 1-chromatic.

In generalizing this concept, a graph G is called p-partite if its vertex set can be decomposed into p disjoint subsets $V_1, V_2, \ldots, V_p$, such that no edge in G joins the vertices in the same subset. Clearly, a κ-chromatic graph is p-partite if and only if

$$\kappa \leq p.$$

With this qualification, the results of this section on κ-chromatic graphs are applicable to κ-partite graphs also.

8-2. CHROMATIC PARTITIONING

A proper coloring of a graph naturally induces a partitioning of the vertices into different subsets. For example, the coloring in Fig. 8-1(c) produces the partitioning

$$\{v_1, v_4\}, \quad \{v_2\}, \quad \text{and} \quad \{v_3, v_5\}.$$

No two vertices in any of these three subsets are adjacent. Such a subset of vertices is called an independent set; more formally:

A set of vertices in a graph is said to be an *independent set* of vertices or simply an *independent set* (or an *internally stable set*) if no two vertices in the set are adjacent. For example, in Fig. 8-3, $\{a, c, d\}$ is an independent set. A single vertex in any graph constitutes an independent set.

A *maximal independent set* (or *maximal internally stable set*) is an independent set to which no other vertex can be added without destroying its independence property. The set $\{a, c, d, f\}$ in Fig. 8-3 is a maximal independent set. The set $\{b, f\}$ is another maximal independent set. The set $\{b, g\}$ is a third one. From the preceding example, it is clear that a graph, in general, has many maximal independent sets; and they may be of different sizes. Among all maximal independent sets, one with the largest number of vertices is often of particular interest.

Suppose that the graph in Fig. 8-3 describes the following problem. Each of the seven vertices of the graph is a possible code word to be used in some communication. Some words are so close (say, in sound) to others that they might be confused for each other. Pairs of such words that may be mistaken for one another are joined by edges. Find a largest set of code words for a reliable communication. This is a problem of finding a maximal independent set with largest number of vertices. In this simple example, $\{a, c, d, f\}$ is an answer.

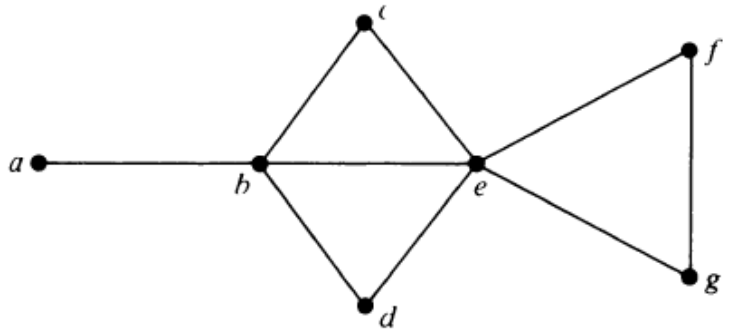


Fig. 8-3

The number of vertices in the largest independent set of a graph G is called the *independence number* (or *coefficient of internal stability*), $\beta(G)$.

Consider a κ-chromatic graph G of n vertices properly colored with κ different colors. Since the largest number of vertices in G with the same color cannot exceed the independence number $\beta(G)$, we have the inequality

$$\beta(G) \geq \frac{n}{\kappa}.$$

Finding a Maximal Independent Set: A reasonable method of finding a maximal independent set in a graph G will be to start with any vertex v of G in the set. Add more vertices to the set, selecting at each stage a vertex that is not adjacent to any of those already selected. This procedure will ultimately produce a maximal independent set. This set, however, is not necessarily a maximal independent set with a largest number of vertices.

Finding All Maximal Independent Sets: A reasonable (but not very efficient for large graphs) method for obtaining all maximal independent sets in any graph can be developed using Boolean arithmetic on the vertices. Let each vertex in the graph be treated as a Boolean variable. Let the logical (or Boolean) sum $a + b$ denote the operation of including vertex a or b or both; let the logical multiplication ab denote the operation of including both vertices a and b, and let the Boolean complement a' denote that vertex a is not included.

For a given graph G we must find a maximal subset of vertices that does not include the two end vertices of any edge in G. Let us express an edge (x, y) as a Boolean product, xy, of its end vertices x and y, and let us sum all such products in G to get a Boolean expression

$$\varphi = \Sigma\, xy \qquad \text{for all } (x, y) \text{ in } G.$$

Let us further take the Boolean complement φ' of this expression, and express it as a sum of Boolean products:

$$\varphi' = f_1 + f_2 + \cdots + f_k.$$

A vertex set is a maximal independent set if and only if $\varphi = 0$ (logically false), which is possible if and only if $\varphi' = 1$ (true), which is possible if and only if at least one $f_i = 1$, which is possible if and only if each vertex appearing in f_i (in complemented form) is excluded from the vertex set of G. Thus each f_i will yield a maximal independent set, and every maximal independent set will be produced by this method. This procedure can be best explained by an example. For the graph G in Fig. 8-3,

$$\varphi = ab + bc + bd + be + ce + de + ef + eg + fg,$$
$$\varphi' = (a' + b')(b' + c')(b' + d')(b' + e')(c' + e')(d' + e')$$
$$(e' + f')(e' + g')(f' + g').$$

Multiplying these out and employing the usual identities of Boolean arithmetic, such as

$$aa = a,$$
$$a + a = a,$$
$$a + ab = a,$$

we get

$$\varphi' = b'e'f' + b'e'g' + a'c'd'e'f' + a'c'd'e'g' + b'c'd'f'g'.$$

Now if we exclude from the vertex set of G vertices appearing in any one of these five terms, we get a maximal independent set. The five maximal independent sets are

$$acdf, \quad acdg, \quad bg, \quad bf, \quad \text{and} \quad ae.$$

These are all the maximal independent sets of the graph.

Finding Independence and Chromatic Numbers: Once all the maximal independent sets of G have been obtained, we find the size of the one with the largest number of vertices to get the independence number $\beta(G)$. The independence number of the graph in Fig. 8-3 is four.

To find the chromatic number of G, we must find the minimum number of these (maximal independent) sets, which collectively include all the vertices of G. For the graph in Fig. 8-3, sets $\{a, c, d, f\}$, $\{b, g\}$, and $\{a, e\}$, for example, satisfy this condition. Thus the graph is 3-chromatic.

Chromatic Partitioning: Given a simple, connected graph G, partition all vertices of G into the smallest possible number of disjoint, independent sets. This problem, known as the *chromatic partitioning* of graphs, is perhaps the most important problem in partitioning of graphs.

By enumerating all maximal independent sets and then selecting the smallest number of sets that include all vertices of the graph, we just solved this problem. The following four are some chromatic partitions of the graph in Fig. 8-3, for example.

$\{(a, c, d, f), (b, g), (e)\}$,
$\{(a, c, d, g), (b, f), (e)\}$,
$\{(c, d, f), (b, g), (a, e)\}$,
$\{(c, d, g), (b, f), (a, e)\}$.

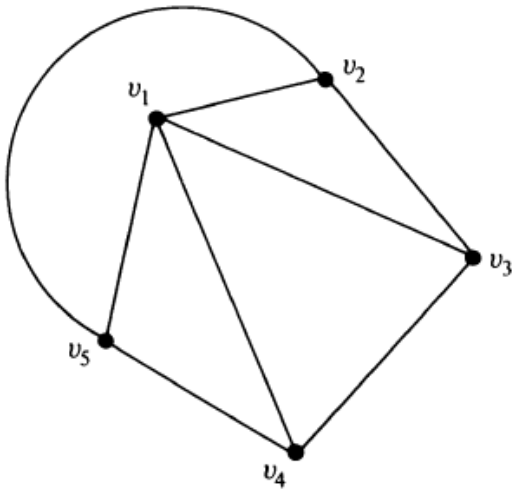


Fig. 8-4 A 3-chromatic graph.

This method of chromatic partitioning (requiring enumeration of all maximal independent sets) is inefficient and needs prohibitively large amounts of computer memory. A more efficient method for computer implementation is proposed in [8-6].

Uniquely Colorable Graphs: A graph that has only one chromatic partition is called a *uniquely colorable* graph. The graph in Fig. 8-3 is *not* a uniquely colorable graph, but the one in Fig. 8-4 is (Problem 8-2). For some interesting properties of uniquely colorable graphs, the reader is referred to Chapter 12 of [1-5].

A concept related to that of the independent set and chromatic partitioning is the dominating set, to be discussed next.

Dominating Sets: A *dominating set* (or an *externally stable set*) in a graph G is a set of vertices that dominates every vertex v in G in the following sense: Either v is included in the dominating set or is adjacent to one or more vertices included in the dominating set. For instance, the vertex set $\{b, g\}$ is a dominating set in Fig. 8-3. So is the set $\{a, b, c, d, f\}$ a dominating set. A dominating set need not be independent. For example, the set of all its vertices is trivially a dominating set in every graph.

In many applications one is interested in finding minimal dominating sets defined as follows:

A *minimal dominating set* is a dominating set from which no vertex can be removed without destroying its dominance property. For example, in Fig. 8-3, $\{b, e\}$ is a minimal dominating set. And so is $\{a, c, d, f\}$. Observations that follow from these definitions are

1. Any one vertex in a complete graph constitutes a minimal dominating set.
2. Every dominating set contains at least one minimal dominating set.
3. A graph may have many minimal dominating sets, and of different sizes. [The number of vertices in the smallest minimal dominating set of a graph G is called the *domination number*, $\alpha(G)$.]
4. A minimal dominating set may or may not be independent.
5. Every maximal independent set is a dominating set. For if an independent set does not dominate the graph, there is at least one vertex that is neither in the set nor adjacent to any vertex in the set. Such a vertex can be added to the independent set without destroying its independence. But then the independent set could not have been maximal.
6. An independent set has the dominance property only if it is a maximal independent set. Thus an *independent dominating set* is the same as a maximal independent set.
7. In any graph G,

$$\alpha(G) \leq \beta(G).$$

Finding Minimal Dominating Sets: A method for obtaining all minimal dominating sets in a graph will now be developed. The method, like the one for finding all maximal independent sets, also uses Boolean arithmetic.

To dominate a vertex v_i we must either include v_i or any of the vertices adjacent to v_i. A minimum set satisfying this condition for every vertex v_i is a desired set. Therefore, for every vertex v_i in G let us form a Boolean product of sums $(v_i + v_{i_1} + v_{i_2} + \cdots + v_{i_d})$, where $v_{i_1}, v_{i_2}, \ldots, v_{i_d}$ are the vertices adjacent to v_i, and d is the degree of v_i:

$$\theta = \prod (v_i + v_{i_1} + v_{i_2} + \cdots + v_{i_d}) \qquad \text{for all } v_i \text{ in } G.$$

When θ is expressed as a sum of products, each term in it will represent a minimal dominating set. Let us illustrate this algorithm using the graph of Fig. 8-3:

Consider the following expression θ for Fig. 8-3:

$$\theta = (a + b)(b + c + d + e + a)(c + b + e)(d + b + e)$$
$$(e + b + c + d + f + g)(f + e + g)(g + e + f).$$

Since in Boolean arithmetic $(x + y)x = x$,

$$\theta = (a + b)(b + c + e)(b + d + e)(e + f + g)$$
$$= ae + be + bf + bg + acdf + acdg.$$

Each of the six terms in the preceding expression represents a minimal dominating set. Clearly, $\alpha(G) = 2$, for this example.

CHROMATIC POLYNOMIAL

In general, a given graph G of n vertices can be properly colored in many different ways using a sufficiently large number of colors. This property of a graph is expressed elegantly by means of a polynomial. This polynomial is called the *chromatic polynomial* of G and is defined as follows:

The value of the chromatic polynomial $P_n(\lambda)$ of a graph with n vertices gives the number of ways of properly coloring the graph, using λ or fewer colors.

Let c_i be the different ways of properly coloring G using exactly i different colors. Since i colors can be chosen out of λ colors in

$$\binom{\lambda}{i} \qquad \text{different ways,}$$

there are $c_i\binom{\lambda}{i}$ different ways of properly coloring G using exactly i colors out of λ colors.

Since i can be any positive integer from 1 to n (it is not possible to use more than n colors on n vertices), the chromatic polynomial is a sum of these terms; that is,

$$c_n = n!.$$

As an illustration, let us find the chromatic polynomial of the graph given in Fig. 8-4.

$$P_5(\lambda) = c_1\lambda + c_2\frac{\lambda(\lambda-1)}{2} + c_3\frac{\lambda(\lambda-1)(\lambda-2)}{3!} + c_4\frac{\lambda(\lambda-1)(\lambda-2)(\lambda-3)}{4!} + c_5\frac{\lambda(\lambda-1)(\lambda-2)(\lambda-3)(\lambda-4)}{5!}.$$

Since the graph in Fig. 8-4 has a triangle, it will require at least three different colors for proper coloring. Therefore,

$$c_1 = c_2 = 0 \quad \text{and} \quad c_5 = 5!.$$

Moreover, to evaluate c_3, suppose that we have three colors x, y, and z. These three colors can be assigned properly to vertices v_1, v_2, and v_3 in $3! = 6$ different ways. Having done that, we have no more choices left, because vertex v_5 must have the same color as v_3, and v_4 must have the same color as v_2. Therefore,

$$c_3 = 6.$$

Similarly, with four colors, v_1, v_2, and v_3 can be properly colored in $4 \cdot 6 = 24$ different ways. The fourth color can be assigned to v_4 or v_5, thus providing two choices. The fifth vertex provides no additional choice. Therefore,

$$c_4 = 24 \cdot 2 = 48.$$

Substituting these coefficients in $P_5(\lambda)$, we get, for the graph in Fig. 8-4,

$$\begin{aligned} P_5(\lambda) &= \lambda(\lambda-1)(\lambda-2) + 2\lambda(\lambda-1)(\lambda-2)(\lambda-3) \\ &\quad + \lambda(\lambda-1)(\lambda-2)(\lambda-3)(\lambda-4) \\ &= \lambda(\lambda-1)(\lambda-2)(\lambda^2-5\lambda+7). \end{aligned}$$

The presence of factors $\lambda - 1$ and $\lambda - 2$ indicates that G is at least 3-chromatic.

Chromatic polynomials have been studied in great detail in the literature. The interested reader is referred to [8-5] for a more thorough discussion of their properties. Theorems 8-4, 8-5, and 8-6 should provide a glimpse into the colorful world of chromatic polynomials.

THEOREM 8-4

A graph of n vertices is a complete graph if and only if its chromatic polynomial is

$$P_n(\lambda) = \lambda(\lambda - 1)(\lambda - 2) \ldots (\lambda - n + 1).$$

Proof: With λ colors, there are λ different ways of coloring any selected vertex of a graph. A second vertex can be colored properly in exactly $\lambda - 1$ ways, the third in $\lambda - 2$ ways, the fourth in $\lambda - 3$ ways, . . . , and the nth in $\lambda - n + 1$ ways if and only if every vertex is adjacent to every other. That is, if and only if the graph is complete. ■

THEOREM 8-5

An n-vertex graph is a tree if and only if its chromatic polynomial

$$P_n(\lambda) = \lambda(\lambda - 1)^{n-1}.$$

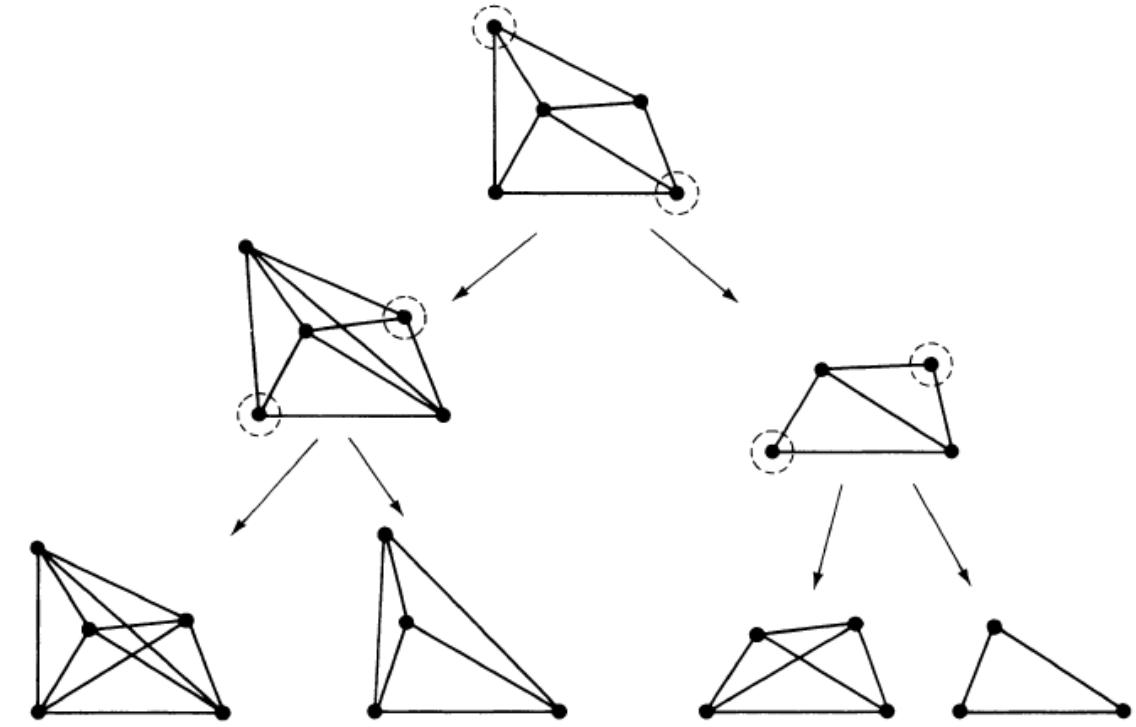
Proof: That the theorem holds for $n = 1, 2$ is immediately evident. It is left as an exercise to prove the theorem by induction (Problem 8-9).

THEOREM 8-6

Let a and b be two nonadjacent vertices in a graph G. Let G' be a graph obtained by adding an edge between a and b. Let G'' be a simple graph obtained from G by fusing the vertices a and b together and replacing sets of parallel edges with single edges. Then

$$P_n(\lambda) \text{ of } G = P_n(\lambda) \text{ of } G' + P_{n-1}(\lambda) \text{ of } G''.$$

Proof: The number of ways of properly coloring G can be grouped into two cases, one such that vertices a and b are of the same color and the other such that a and b are of different colors. Since the number of ways of properly coloring G such that a and b have different colors = number of ways of properly coloring G', and
number of ways of properly coloring G such that a and b have the same color



$$P_5(\lambda) \text{ of } G = \lambda(\lambda - 1)(\lambda - 2) + 2\lambda(\lambda - 1)(\lambda - 2)(\lambda - 3)$$
$$+ \lambda(\lambda - 1)(\lambda - 2)(\lambda - 3)(\lambda - 4)$$
$$= \lambda(\lambda - 1)(\lambda - 2)(\lambda^2 - 5\lambda + 7)$$

Fig. 8-5 Evaluation of a chromatic polynomial.

= number of ways of properly coloring G'',

$$P_n(\lambda) \text{ of } G = P_n(\lambda) \text{ of } G' + P_{n-1}(\lambda) \text{ of } G''. \quad \blacksquare$$

Theorem 8-6 is often used in evaluating the chromatic polynomial of a graph. For example, Fig. 8-5 illustrates how the chromatic polynomial of a graph G is expressed as a sum of the chromatic polynomials of four complete graphs. The pair of nonadjacent vertices shown enclosed in circles is the one used for reduction at that stage.

In the last three sections we have been concerned with proper coloring of the vertices in a graph. Suppose that we are interested in coloring the edges rather than the vertices. It is reasonable to call two edges *adjacent* if they have one end vertex in common (but are not parallel). A proper coloring of edges then requires that adjacent edges should be of different colors. Some results on proper coloring of edges, similar to the results given in Sections 8-1 and 8-2, can be derived (Problem 8-19).

Moreover, a set of edges in which no two are adjacent is similar to an independent set of vertices. Such a set of edges is called a *matching*, the subject of the next section.

MATCHINGS

Suppose that four applicants a_1, a_2, a_3, and a_4 are available to fill six vacant positions p_1, p_2, p_3, p_4, p_5, and p_6. Applicant a_1 is qualified to fill position p_2 or p_5. Applicant a_2 can fill p_2 or p_5. Applicant a_3 is qualified for p_1, p_2, p_3, p_4, or p_6. Applicant a_4 can fill jobs p_2 or p_5. This situation is represented by the graph in Fig. 8-6. The vacant positions and applicants are represented by vertices. The edges represent the qualifications of each applicant for filling

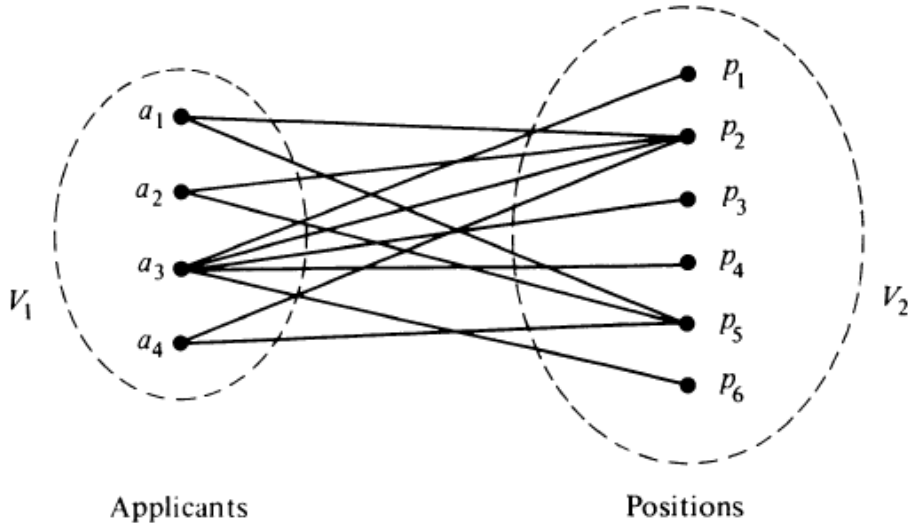


Fig. 8-6 Bipartite graph.

Fig. 8-7 Graph and two of its maximal matchings.

different positions. The graph clearly is bipartite, the vertices falling into two sets $V_1 = \{a_1, a_2, a_3, a_4\}$ and $V_2 = \{p_1, p_2, p_3, p_4, p_5, p_6\}$.

The questions one is most likely to ask in this situation are: Is it possible to hire all the applicants and assign each a position for which he is suitable? If the answer is no, what is the maximum number of positions that can be filled from the given set of applicants?

This is a problem of *matching* (or *assignment*) of one set of vertices into another. More formally, a *matching* in a graph is a subset of edges in which no two edges are adjacent. A single edge in a graph is obviously a matching.

A *maximal matching* is a matching to which no edge in the graph can be added. For example, in a complete graph of three vertices (i.e., a triangle) any single edge is a maximal matching. The edges shown by heavy lines in Fig. 8-7 are two maximal matchings. Clearly, a graph may have many different maximal matchings, and of different sizes. Among these, the maximal *matchings*. In Fig. 8-7(b), a largest maximal matching is shown in heavy lines. The number of edges in a largest maximal matching is called the *matching number* of the graph.

Although matching is defined for any graph, it is mostly studied in the context of bipartite graphs, as suggested by the introduction to this section. In a bipartite graph having a vertex partition V_1 and V_2, a *complete matching* of vertices in set V_1 into those in V_2 is a matching in which there is one edge incident with every vertex in V_1. In other words, every vertex in V_1 is matched against some vertex in V_2. Clearly, a complete matching (if it exists) is a largest maximal matching, whereas the converse is not necessarily true.

For the existence of a complete matching of set V_1 into set V_2, first we must have at least as many vertices in V_2 as there are in V_1. In other words, there must be at least as many vacant positions as the number of applicants if all the applicants are to be hired. This condition, however, is not sufficient. For example, in Fig. 8-6, although there are six positions and four applicants, a complete matching does not exist. Of the three applicants a_1, a_2, and a_4, each qualifies for the same two positions p_2 and p_5, and therefore one of the three applicants cannot be matched.

This leads us to another necessary condition for a complete matching: Every subset of r vertices in V_1 must collectively be adjacent to at least r vertices in V_2, for all values of $r = 1, 2, \ldots, |V_1|$. This condition is not satisfied in Fig. 8-6. The subset $\{a_1, a_2, a_4\}$ of three vertices has only two vertices p_2 and p_5 adjacent to them. That this condition is also sufficient for existence of a complete matching is indeed surprising. Theorem 8-7 is a formal statement and proof of this result.

THEOREM 8-7

A complete matching of V_1 into V_2 in a bipartite graph exists if and only if every subset of r vertices in V_1 is collectively adjacent to r or more vertices in V_2 for all values of r.

Proof: The "only if" part (i.e., the necessity of a subset of r applicants collectively qualifying for at least r jobs) is immediate and has already been pointed out. The sufficiency (i.e., the "if" part) can be proved by induction on r, as the theorem trivially holds for $r = 1$. For a complete proof, the student is referred to Theorem 11-1 in [8-3], Theorem 5-19 in [4-5], or Chapter 4 in [1-9].

Problem of Distinct Representatives: Five senators s_1, s_2, s_3, s_4, and s_5 are members of three committees, c_1, c_2, and c_3. The membership is shown in Fig. 8-8. One member from each committee is to be represented in a supercommittee. Is it possible to send one distinct representative from each of the committees†?

This problem is one of finding a complete matching of a set V_1 into set V_2 in a bipartite graph. Let us use Theorem 8-7 and check if r vertices from V_1 are collectively adjacent to at least r vertices from V_2, for all values of r. The result is shown in Table 8-1 (ignore the last column for the time being).

Thus for this example the condition for the existence of a complete matching is satisfied as stated in Theorem 8-7. Hence it is possible to form the supercommittee with one distinct representative from each committee.

The problem of distinct representatives just solved was a small one. A

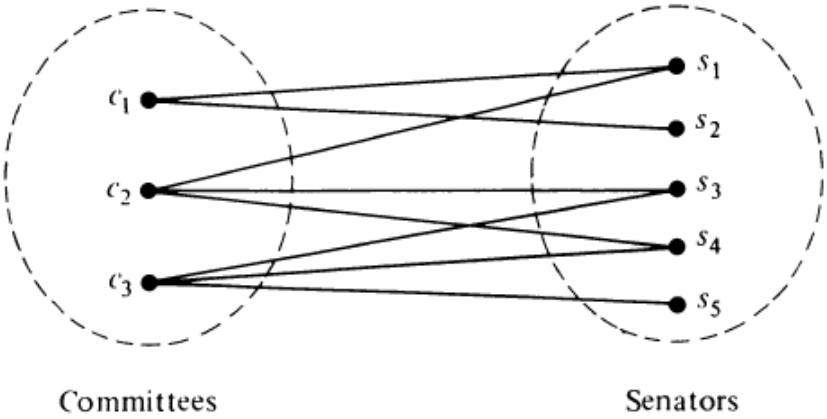


Fig. 8-8 Membership of committees.

	V_1	V_2	$r - q$
$r = 1$	$\{c_1\}$	$\{s_1, s_2\}$	-1
	$\{c_2\}$	$\{s_1, s_3, s_4\}$	-2
	$\{c_3\}$	$\{s_3, s_4, s_5\}$	-2
$r = 2$	$\{c_1, c_2\}$	$\{s_1, s_2, s_3, s_4\}$	-2
	$\{c_2, c_3\}$	$\{s_1, s_3, s_4, s_5\}$	-2
	$\{c_3, c_1\}$	$\{s_1, s_2, s_3, s_4, s_5\}$	-3
$r = 3$	$\{c_1, c_2, c_3\}$	$\{s_1, s_2, s_3, s_4, s_5\}$	-2

Table 8-1

larger problem would have become unwieldy. If there are M vertices in V_1, Theorem 8-7 requires that we take all $2^M - 1$ nonempty subsets of V_1 and find the number of vertices of V_2 adjacent collectively to each of these. In most cases, however, the following simplified version of Theorem 8-7 will suffice for detection of a complete matching in any large graph.

THEOREM 8-8

In a bipartite graph a complete matching of V_1 into V_2 exists if (but not only if) there is a positive integer m for which the following condition is satisfied:

degree of every vertex in $V_1 \geq m \geq$ degree of every vertex in V_2.

Proof: Consider a subset of r vertices in V_1. These r vertices have at least $m \cdot r$ edges incident on them. Each $m \cdot r$ edge is incident to some vertex in V_2. Since the degree of every vertex in set V_2 is no greater than m, these $m \cdot r$ edges are incident on at least $(m \cdot r)/m = r$ vertices in V_2.

Thus any subset of r vertices in V_1 is collectively adjacent to r or more vertices in V_2. Therefore, according to Theorem 8-7, there exists a complete matching of V_1 into V_2. ■

In the bipartite graph of Fig. 8-8,

degree of every vertex in $V_1 \geq 2 \geq$ degree of every vertex in V_2.

Therefore, there exists a complete matching.

In the bipartite graph of Fig. 8-6 no such number is found, because the degree of $p_2 = 4 >$ degree of a_1.

It must be emphasized that the condition of Theorem 8-8 is a sufficient condition and not necessary for the existence of a complete matching. It will be instructive for the reader to sketch a bipartite graph that does not satisfy Theorem 8-8 and yet has a complete matching (Problem 8-15).

The matching problem or the problem of distinct representatives is also called the *marriage problem* (whose solution, unfortunately, is of little use to those with real marital problems!) See Problem 8-16.

If one fails to find a complete matching, he is most likely to be interested in finding a maximal matching, that is, to pair off as many vertices of V_1 with those in V_2 as possible. For this purpose, let us define a new term called *deficiency*, $\delta(G)$, of a bipartite graph G.

A set of r vertices in V_1 is collectively incident on, say, q vertices of V_2. Then the maximum value of the number $r - q$ taken over all values of $r = 1, 2, \ldots$ and all subsets of V_1 is called the deficiency $\delta(G)$ of the bipartite graph G.

Theorem 8-7, expressed in terms of the deficiency, states that a complete matching in a bipartite graph G exists if and only if

$$\delta(G) \leq 0.$$

For example, the deficiency of the bipartite graph in Fig. 8-7 is -1 (the largest number in the last column of Table 8-1). It is suggested that you prepare a table for the graph of Fig. 8-6, similar to Table 8-1, and verify that the deficiency is $+1$ for this graph (Problem 8-17).

Theorem 8-9 gives the size of the maximal matching for a bipartite graph with a positive deficiency.

THEOREM 8-9

The maximal number of vertices in set V_1 that can be matched into V_2 is equal to

$$\text{number of vertices in } V_1 - \delta(G).$$

The proof of Theorem 8-9 can be found in [8-3], page 288. The size of a maximal matching in Fig. 8-6, using Theorem 8-9, is obtained as follows:

$$\text{number of vertices in } V_1 - \delta(G) = 4 - 1 = 3.$$

Matching and Adjacency Matrix: Consider a bipartite graph G with non-adjacent sets of vertices V_1 and V_2, having number of vertices n_1 and n_2, respectively, and let $n_1 \leq n_2$, $n_1 + n_2 = n$, the number of vertices in G. The adjacency matrix $\mathsf{X}(G)$ of G can be written in the form

$$\mathsf{X}(G) = \left[\begin{array}{c|c} 0 & \mathsf{X}_{12} \\ \hline \mathsf{X}_{12}^T & 0 \end{array}\right],$$

where the submatrix X_{12} is the n_1 by n_2, (0, 1)-matrix containing the information as to which of the n_1 vertices of V_1 are connected to which of the n_2 vertices of V_2. Matrix X_{12}^T is the transpose of X_{12}.

Clearly, all the information about the bipartite graph G is contained in its X_{12} matrix.

A matching V_1 into V_2 corresponds to a selection of the 1's in the matrix X_{12} such that no line (i.e., a row or a column) has more than one 1.

The matching is complete if the n_1 by n_2 matrix made of selected 1's has exactly one 1 in every row. For example, the X_{12} matrix for Fig. 8-8 is

$$\mathsf{X}_{12} = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \end{array} \begin{array}{c} \begin{array}{ccccc} s_1 & s_2 & s_3 & s_4 & s_5 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix} \end{array},$$

$$n_1 = 3, \quad n_2 = 5, \quad n = 8, \quad \text{and} \quad n_1 \leq n_2,$$

$$V_1 = \{c_1, c_2, c_3\}$$

$$V_2 = \{s_1, s_2, s_3, s_4, s_5\}.$$

A complete matching of V_1 into V_2 is given by

$$\mathsf{M} = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \end{array} \begin{array}{c} \begin{array}{ccccc} s_1 & s_2 & s_3 & s_4 & s_5 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}.$$

A maximal matching corresponds to the selection of a largest possible number of 1's from X_{12} such that no row in it has more than one 1. Therefore, according to Theorem 8-9, in matrix X_{12} the largest number of 1's, no two of which are in one row, is equal to

$$\text{number of vertices in } V_1 - \delta(G).$$

Matching problems in bipartite graphs can also be formulated in terms of the flow problem (see Section 14-5). All edges are assumed to be of unit capacity, and the problem of finding a maximal matching is reduced to the problem of maximizing flow from the source to the sink (also see [8-3]).

COVERINGS

In a graph G, a set g of edges is said to *cover* G if every vertex in G is incident on at least one edge in g. A set of edges that covers a graph G is said to be an *edge covering*, a *covering subgraph*, or simply a *covering* of G. For example, a graph G is trivially its own covering. A spanning tree in a connected graph (or a spanning forest in an unconnected graph) is another covering. A Hamiltonian circuit (if it exists) in a graph is also a covering.

Just any covering is too general to be of much interest. We have already dealt with some coverings with specific properties, such as spanning trees and

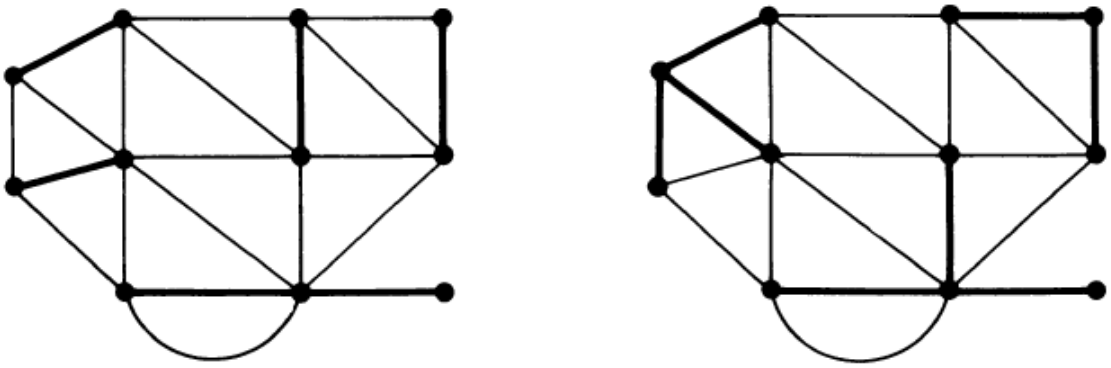


Fig. 8-9 Graph and two of its minimal coverings.

Hamiltonian circuits. In this section we shall investigate the *minimal covering*—a covering from which no edge can be removed without destroying its ability to cover the graph. In Fig. 8-9 a graph and two of its minimal coverings are shown in heavy lines.

The following observations should be made:

1. A covering exists for a graph if and only if the graph has no isolated vertex.
2. A covering of an n-vertex graph will have at least $\lceil n/2 \rceil$ edges. ($\lceil x \rceil$ denotes the smallest integer not less than x.)
3. Every pendant edge in a graph is included in every covering of the graph.
4. Every covering contains a minimal covering.
5. If we denote the remaining edges of a graph by $(G - g)$, the set of edges g is a covering if and only if, for every vertex v, the degree of vertex in $(G - g) \leq$ (degree of vertex v in G) $- 1$.
6. No minimal covering can contain a circuit, for we can always remove an edge from a circuit without leaving any of the vertices in the circuit uncovered. Therefore, a minimal covering of an n-vertex graph can contain no more than $n - 1$ edges.

7. A graph, in general, has many minimal coverings, and they may be of different sizes (i.e., consisting of different numbers of edges). The number of edges in a minimal covering of the smallest size is called the *covering number* of the graph.

THEOREM 8-10

A covering g of a graph is minimal if and only if g contains no paths of length three or more.

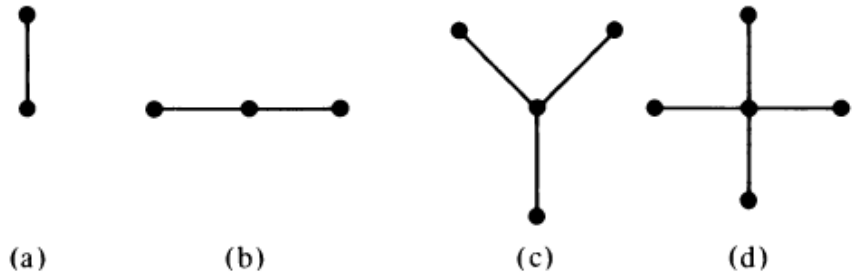


Fig. 8-10 Star graphs of one, two, three, and four edges.

Proof: Suppose that a covering g contains a path of length three, and it is

$$v_1e_1v_2e_2v_3e_3v_4.$$

Edge e_2 can be removed without leaving its end vertices v_2 and v_3 uncovered. Therefore, g is not a minimal covering.

Conversely, if a covering g contains no path of length three or more, all its components must be *star graphs* (i.e., graphs in the shape of stars; see Fig. 8-10). From a star graph no edge can be removed without leaving a vertex uncovered. That is, g must be a minimal covering. ■

Suppose that the graph in Fig. 8-9 represents the street map of a part of a city. Each of the vertices is a potential trouble spot and must be kept under the surveillance of a patrol car. How will you assign a minimum number of patrol cars to keep every vertex covered?

The answer is a smallest minimal covering. The covering shown in Fig. 8-9(a) is an answer, and it requires six patrol cars. Clearly, since there are 11 vertices and no edge can cover more than two, less than six edges cannot cover the graph.

Minimization of Switching Functions†: An important step in the logical design of a digital machine is to minimize Boolean functions before implementing them. Suppose we are interested in building a logical circuit that gives the following function F of four Boolean variables w, x, y, and z.

$$F = \bar{w}\bar{x}\bar{y}\bar{z} + \bar{w}\bar{x}y\bar{z} + w\bar{x}\bar{y}\bar{z} + \bar{w}\bar{x}yz + \bar{w}xy\bar{z} + \bar{w}xyz + wxyz,$$

where $+$ denotes logical OR, xy denotes x AND y, and $\bar{x}$ denotes NOT x.

Let us represent each of the seven terms in F by a vertex, and join every pair of vertices that differ only in one variable. Such a graph is shown in Fig. 8-11.

An edge between two vertices represents a term with three variables.

A minimal cover of this graph will represent a simplified form of F, performing the same function as F, but with less logic hardware.

The pendant edges 1 and 7 must be included in every covering of the

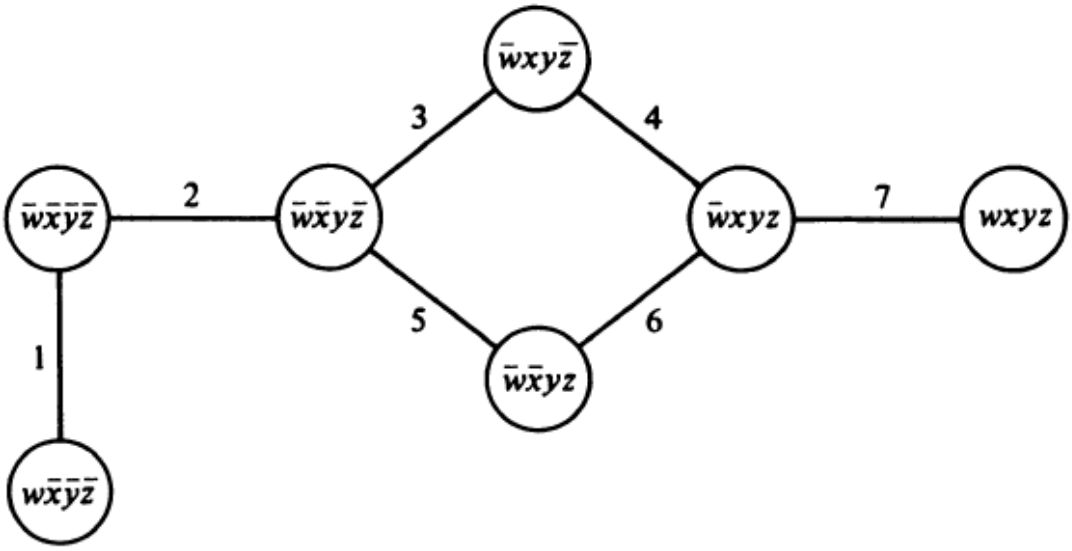


Fig. 8-11 Graph representation of a Boolean function.

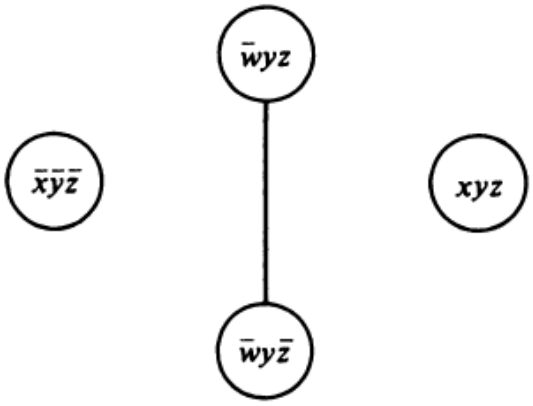


Fig. 8-12

graph. Therefore, the terms

$$\bar{x}\bar{y}\bar{z} \quad \text{and} \quad xyz \quad \text{are essential.}$$

Two additional edges 3 and 6 (or 4 and 5 or 3 and 5) will cover the remainder. Thus a simplified version of F is

$$F = \bar{x}\bar{y}\bar{z} + xyz + \bar{w}y\bar{z} + \bar{w}yz.$$

This expression can again be represented by a graph of four vertices, as shown in Fig. 8-12.

The essential terms $\bar{x}\bar{y}\bar{z}$ and xyz cannot be covered by any edge, and hence cannot be minimized further. One edge will cover the remaining two vertices in Fig. 8-12. Thus the minimized Boolean expression is

$$F = \bar{x}\bar{y}\bar{z} + xyz + \bar{w}y.$$

Dimer Problem: In crystal physics, a crystal is represented by a three-dimensional lattice. Each vertex in the lattice represents an atom, and an edge between vertices represents the bond between the two atoms. In the study of the surface properties of crystals, one is interested in two-dimensional lattices, such as the two shown in Fig. 1-10.

To obtain an analytic expression for certain surface properties of crystals consisting of diatomic molecules (also called *dimers*), one is required to find the number of ways in which all atoms on a two-dimensional lattice can be paired off as molecules (each consisting of two atoms). The problem is equiv-

alent to finding all different coverings of a given graph such that every vertex in the covering is of degree one. Such a covering in which every vertex is of degree one is called a *dimer covering* or a *1-factor*. A dimer covering is obviously a matching because no two edges in it are adjacent. Moreover, a dimer covering is a maximal matching. This is why a dimer covering is often referred to as a *perfect matching*.

Two different dimer coverings are shown in heavy lines in the graph in Fig. 8-13.

Clearly, a graph must have an even number of vertices to have a dimer covering. This condition, however, is not enough (Problem 8-21).

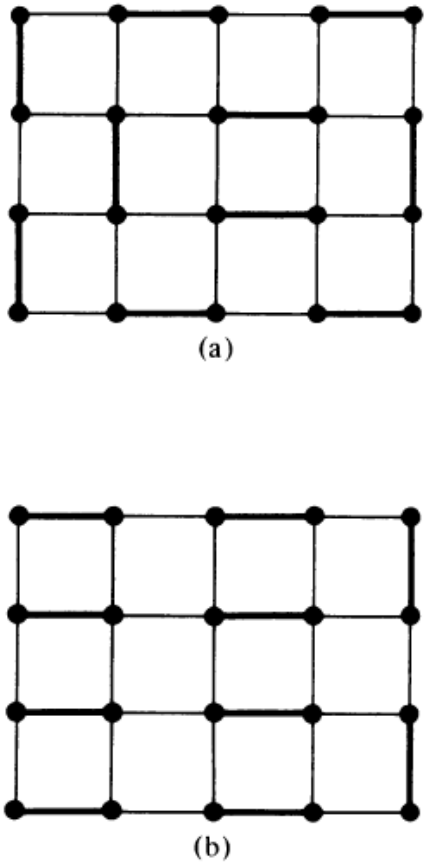


Fig. 8-13 Two dimer coverings of a graph.

FOUR-COLOR PROBLEM

So far we have considered proper coloring of vertices and proper coloring of edges. Let us briefly consider the *proper coloring of regions* in a planar graph (embedded on a plane or sphere). Just as in coloring of vertices and

edges, the regions of a planar graph are said to be properly colored if no two *contiguous* or *adjacent regions* have the same color. (Two regions are said to be adjacent if they have a common edge between them. Note that one or more vertices in common does not make two regions adjacent.) The proper coloring of regions is also called *map coloring*, referring to the fact that in an atlas different countries are colored such that countries with common boundaries are shown in different colors.

Once again we are not interested in just properly coloring the regions of a given graph. We are interested in a coloring that uses the minimum number of colors. This leads us to the most famous conjecture in graph theory. The conjecture is that every map (i.e., a planar graph) can be properly colored with four colors. The *four-color conjecture*, already referred to in Chapter 1, has been worked on by many famous mathematicians for the past 100 years. No one has yet been able to either prove the theorem or come up with a map (in a plane) that requires more than four colors.

That at least four colors are necessary to properly color a graph is immediate from Fig. 8-14, and that five colors will suffice for any planar graph will be shown shortly.

Two remarks may be made here in passing. Paradoxically, for surfaces more complicated than the plane (or sphere) corresponding theorems have been proved. For example, it has been proved that seven colors are necessary and sufficient for properly coloring maps on the surface of a torus.† Second, it has been proved that all maps containing less than 40 regions can be properly colored with four colors. Therefore, if in general the four-color conjecture is false, the counterexample has to be a very complicated and large one.

Vertex Coloring Versus Region Coloring: From Chapter 5 we know that a graph has a dual if and only if it is planar. Therefore, coloring the regions of a planar graph G is equivalent to coloring the vertices of its dual G^*, and

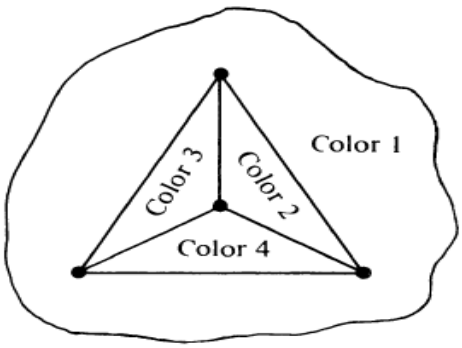


Fig. 8-14 Necessity of four colors.

vice versa. Thus the four-color conjecture can be restated as follows: *Every planar graph has a chromatic number of four or less.*

Five-Color Theorem: We shall now show that every planar map can be properly colored with five colors:

THEOREM 8-11

The vertices of every planar graph can be properly colored with five colors.

Proof: The theorem will be proved by induction. Since the vertices of all graphs (self-loop-free, of course†) with 1, 2, 3, 4, or 5 vertices can be properly colored with five colors, let us assume that vertices of every planar graph with $n - 1$ vertices can be properly colored with five colors. Then, if we prove that any planar graph G with n vertices will require no more than five colors, we shall have proved the theorem.

Consider the planar graph G with n vertices. Since G is planar, it must have at least one vertex with degree five or less (Problem 5-4). Let this vertex be v.

Let G' be a graph (of $n - 1$ vertices) obtained from G by deleting vertex v (i.e., v and all edges incident on v). Graph G' requires no more than five colors, according to the induction hypothesis. Suppose that the vertices in G' have been properly colored, and now we add to it v and all edges incident on v. If the degree of v is 1, 2, 3, or 4, we have no difficulty in assigning a proper color to v.

1, 2, 3, or 4, we have no difficulty in assigning a proper color to v.

This leaves only the case in which the degree of v is five, and all the five colors have been used in coloring the vertices adjacent to v, as shown in Fig. 8-15(a). (Note that Fig. 8-15 is part of a planar representation of graph G'.)

Suppose that there is a path in G' between vertices a and c colored alternately with colors 1 and 3, as shown in Fig. 8-15(b). Then a similar path between b and d, colored alternately with colors 2 and 4, cannot exist; otherwise, these two paths

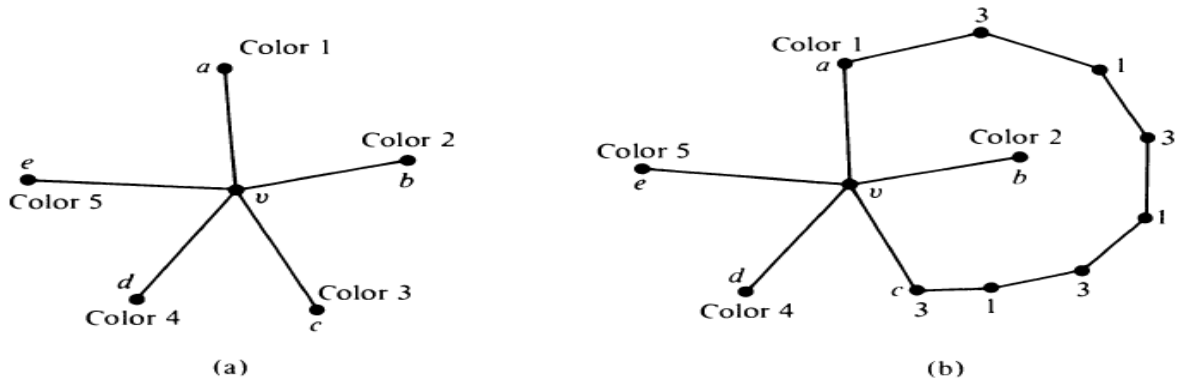


Fig. 8-15 Reassigning of colors.

will intersect and cause G to be nonplanar. (This is a consequence of the Jordan curve theorem, used in Section 5-3, also.)

If there is no path between b and d colored alternately with colors 2 and 4, starting from vertex b we can interchange colors 2 and 4 of all vertices connected to b through vertices of alternating colors 2 and 4. This interchange will paint vertex b with color 4 and yet keep G' properly colored. Since vertex d is still with color 4, we have color 2 left over with which to paint vertex v.

Had we assumed that there was no path between a and c of vertices painted alternately with colors 1 and 3, we would have released color 1 or 3 instead of color 2. And thus the theorem. ■

Regularization of a Planar Graph: Removing every vertex of degree one (together with the pendant edge) from the graph G does not affect the regions of a planar graph. Nor does the elimination of every vertex of degree two, by merging the two edges in series (Fig. 5-6), have any effect on the regions of a planar graph.

Now consider a typical vertex v of degree four or more in a planar graph. Let us replace vertex v by a small circle with as many vertices as there were incidences on v. This results in a number of vertices each of degree three (see Fig. 8-16).

Performing this transformation on every vertex of degree four or more in a planar graph G will produce another planar graph H in which every vertex is of degree three. When the regions of H have been properly colored, a proper coloring of the regions of G can be obtained simply by shrinking each of the new regions back to the original vertex.

Such a transformation may be called *regularization* of a planar graph, because it converts a planar graph G into a regular planar graph H of degree three. Clearly, if H can be colored with four colors, so can G. Thus, for map-coloring problems, it is sufficient to confine oneself to (connected) planar, regular graphs of degree three. And the four-color conjecture may be restated as follows:

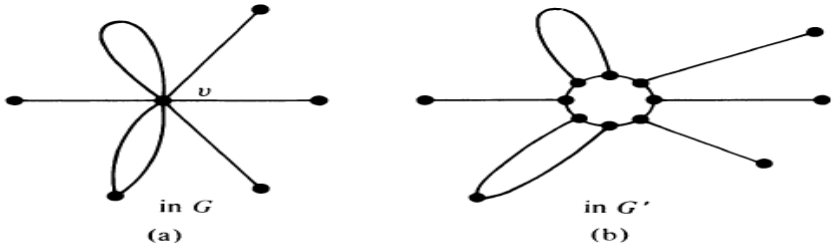


Fig. 8-16 Regularization of a graph.

The regions of every planar, regular graph of degree three can be colored properly with four colors.

If, in a planar graph G, every vertex is of degree three, its dual G^* is a planar graph in which every region is bounded by three edges; that is, G^* is a triangular graph. Thus the four-color conjecture may again be restated as follows: The chromatic number of every triangular, planar graph is four or less.

POSSIBLE QUESTIONS

2 Mark Questions:

1. Define Mobius Inversion Formula.
2. Prove that the function μ is a multiplicative function.
3. Define greatest positive integer.

4. If N is a positive integer, then $\sum_{n=1}^{N}\tau(n)=\sum_{n=1}^{N}[N/n]$.
5. Define Euler Phi function with example.
6. Find the value of $\phi(36000)$.
7. Prove that for $n>2$, $\phi(n)$ is an even integer.
8. Prove that for any positive integer n, $\phi(n)=n\sum_{d|n}\mu(d)/d$.
9. State Gauss lemma.
10. Prove that if p is a prime and p does not divides a,then $a^{p-1}\equiv 1(\text{mod } p)$.

8 Mark Questions:

1. State and prove Mobius inverse formula.
2. Prove that if F is multiplicative function

$$F(n)=\sum_{d|n}f(d),$$

Then f is also multiplicative.

3. Prove that if n is a positive integer and p is a prime, then the exponent of the highest power of p that divides n! is

$$\sum_{n=1}^{\infty}[n/pk]$$

(That is an infinite series, since $[n/p^k]=0$ for $p^k>$n.)

4. Prove if n and r are positive integers with $1\le r<n$, then the binomial coefficient

$$\binom{n}{r}=\frac{n!}{r!(n-r)!}$$

is also an integer.

5. Let f and F be number-theoretic function such that

$$F(n)=\sum_{d|n}f(d),$$

then, prove for any positive integer N,

$$\sum_{k=1}^{N}F(n)=\sum_{k=1}^{N}f(k)[N/k].$$

6. Prove that the function ϕ is a multiplicative function.
7. Prove that if the integer $n>1$has the prime factorization $n=p_1^{k_1}p_2^{k_2}\ldots p_r^{k_r}$, then

$$\phi(n)=(p_1^{k_1}-p_1^{k_1-1})(p_2^{k_2}-p_2^{k_2-1})\ldots(p_r^{k_r}-p_r^{k_r-1})$$
$$=n(1-1/p_1)(1-1/p_2)\ldots(1-1/p_r).$$

8. Let $n>1$ and $\gcd(a,n)=1$. If $a_1,a_2,...,a_{\phi(n)}$ are the positive integer less than n and relatively prime to n, then

$$aa_1, aa_2, ..., aa_{\phi(n)}$$

are congruent modulo n to $a_1,a_2,...,a_{\phi(n)}$ in some order.

9. State and prove Euler theorem.
10. Prove that for each positive integer $n \geq 1$,

$$n=\sum_{d|n}\phi(d),$$

the sum being extended over all positive divisor of n.

UNIT-IV

SYLLABUS

Directed Graphs – Types of Directed Graphs - Types of enumeration, counting labeled trees, counting unlabelled trees, Polya's counting theorem, graph enumeration with Polya's theorem.

DIRECTED GRAPH

A *directed graph* (or a *digraph* for short) G consists of a set of vertices $V = \{v_1, v_2, \ldots\}$, a set of edges $E = \{e_1, e_2, \ldots\}$, and a mapping Ψ that maps

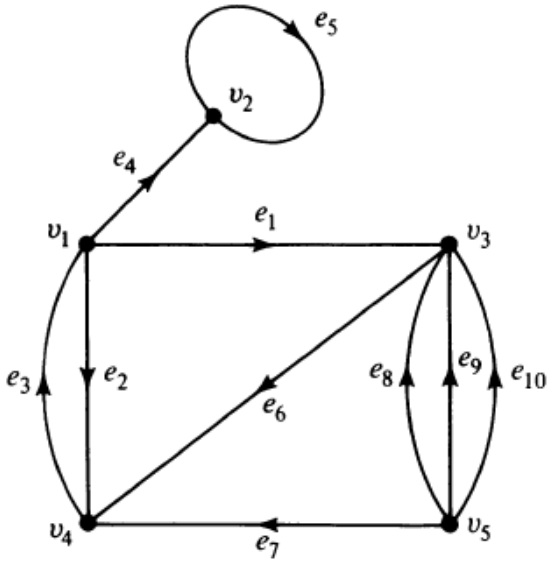


Fig. 9-1 Directed graph with 5 vertices and 10 edges.

every edge onto some *ordered* pair of vertices (v_i, v_j). As in the case of undirected graphs, a vertex is represented by a point and an edge by a line segment between v_i and v_j with an arrow directed from v_i to v_j. For example, Fig. 9-1 shows a digraph with five vertices and ten edges. A digraph is also referred to as an *oriented graph*.†

In a digraph an edge is not only incident on a vertex, but is also *incident out of* a vertex and *incident into* a vertex. The vertex v_i, which edge e_k is incident out of, is called the *initial vertex* of e_k. The vertex v_j, which e_k is incident into, is called the *terminal vertex* of e_k. In Fig. 9-1, v_5 is the initial vertex and v_4 is the terminal vertex of edge e_7. An edge for which the initial and terminal vertices are the same forms a *self-loop*, such as e_5. (Some authors reserve the term *arc* for an oriented or directed edge. We use the term edge to mean either an undirected or a directed edge. Whenever there is a possibility of confusion, we shall explicitly state directed or undirected edge.)

The number of edges incident out of a vertex v_i is called the *out-degree* (or *out-valence* or *outward demidegree*) of v_i and is written $d^+(v_i)$. The number of edges incident into v_i is called the *in-degree* (or *in-valence* or *inward demi-degree*) of v_i and is written as $d^-(v_i)$. In Fig. 9-1, for example,

$$d^+(v_1) = 3, \qquad d^-(v_1) = 1,$$
$$d^+(v_2) = 1, \qquad d^-(v_2) = 2,$$
$$d^+(v_5) = 4, \qquad d^-(v_5) = 0.$$

It is not difficult to prove (Problem 9-1) that in any digraph G the sum of all in-degrees is equal to the sum of all out-degrees, each sum being equal to the number of edges in G; that is,

$$\sum_{i=1}^{n} d^+(v_i) = \sum_{i=1}^{n} d^-(v_i).$$

An *isolated vertex* is a vertex in which the in-degree and the out-degree are both equal to zero. A vertex v in a digraph is called *pendant* if it is of degree one, that is, if

$$d^+(v) + d^-(v) = 1.$$

Two directed edges are said to be *parallel* if they are mapped onto the same ordered pair of vertices. That is, in addition to being parallel in the sense of undirected edges, parallel directed edges must also agree in the direction of their arrows. In Fig. 9-1, edges e_8, e_9, and e_{10} are parallel, whereas edges e_2 and e_3 are not.

Since many properties of directed graphs are the same as those of undirected ones, it is often convenient to disregard the orientations of edges in a digraph. Such an undirected graph obtained from a directed graph G will be called the *undirected graph corresponding to G*.

On the other hand, given an undirected graph H, we can assign each edge of H some arbitrary direction. The resulting digraph, designated by $\vec{H}$ is called an *orientation of H* (or a *digraph associated with H*). Note that while a given digraph has a unique (within isomorphism) undirected graph corresponding to it, a given undirected graph may have "different" orientations possible. This is why we say *the* undirected graph corresponding to a digraph, but *an* orientation of a graph.

Isomorphic Digraphs: Isomorphic graphs were defined such that they have identical behavior in terms of graph properties. In other words, if their labels are removed, two isomorphic graphs are indistinguishable. For two digraphs

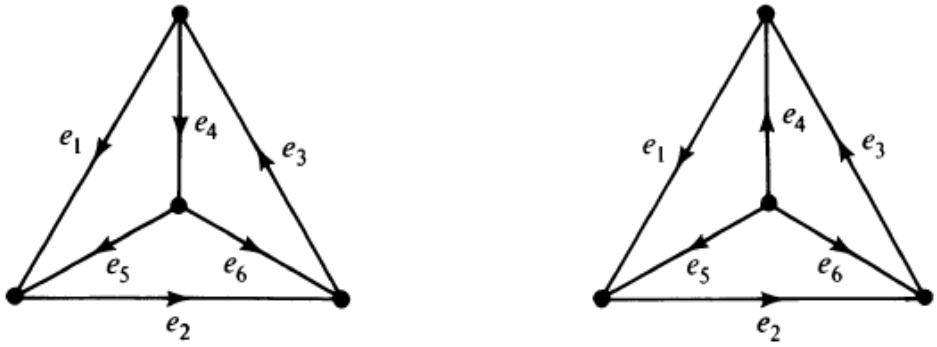


Fig. 9-2 Two nonisomorphic digraphs.

to be isomorphic not only must their corresponding undirected graphs be isomorphic, but the directions of the corresponding edges must also agree. For example, Fig. 9-2 shows two digraphs that are not isomorphic, although they are orientations of the same undirected graph.

Figure 9-2 immediately suggests a problem. What is the number of distinct (i.e., nonisomorphic) orientations of a given undirected graph? The problem was solved by F. Harary and E. M. Palmer in 1966. Some specific cases are left as an exercise (Problem 9-3).

SOME TYPES OF DIGRAPHS

Like their undirected sisters, digraphs come in many varieties. In fact, due to the choice of assigning a direction to each edge, directed graphs have more varieties than undirected ones.

Simple Digraphs: A digraph that has no self-loop or parallel edges is called a simple digraph (Figs. 9-2 and 9-3, for example).

Asymmetric Digraphs: Digraphs that have at most one directed edge between a pair of vertices, but are allowed to have self-loops, are called *asymmetric* or *antisymmetric*.

Symmetric Digraphs: Digraphs in which for every edge (a, b) (i.e., from vertex a to b) there is also an edge (b, a).

A digraph that is both simple and symmetric is called a *simple symmetric digraph*. Similarly, a digraph that is both simple and asymmetric is *simple asymmetric*. The reason for the terms symmetric and asymmetric will be apparent in the context of binary relations in Section 9-3.

Complete Digraphs: A complete undirected graph was defined as a simple graph in which every vertex is joined to every other vertex exactly by one edge. For digraphs we have two types of complete graphs. A *complete symmetric digraph* is a simple digraph in which there is exactly one edge directed from every vertex to every other vertex (Fig. 9-3), and a *complete asymmetric digraph* is an asymmetric digraph in which there is exactly one edge between every pair of vertices (Fig. 9-2).

A complete asymmetric digraph of n vertices contains $n(n-1)/2$ edges, but a complete symmetric digraph of n vertices contains $n(n-1)$ edges. A complete asymmetric digraph is also called a *tournament* or a *complete tournament* (the reason for this term will be made clear in Section 9-10).

A digraph is said to be *balanced* if for every vertex v_i the in-degree equals the out-degree; that is, $d^+(v_i) = d^-(v_i)$. (A balanced digraph is also referred to as a *pseudosymmetric* digraph, or an *isograph*.) A balanced digraph is said to be *regular* if every vertex has the same in-degree and out-degree as every other vertex.

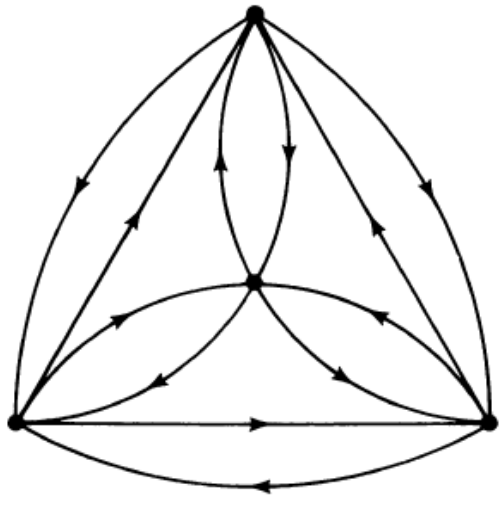


Fig. 9-3 Complete symmetric digraph of four vertices.

DIGRAPHS AND BINARY RELATIONS

The theory of graphs and the calculus of binary relations are closely related (so much so that some people often mistakenly come to regard graph theory as a branch of the theory of relations).

In a set of objects, X, where

$$X = \{x_1, x_2, \ldots\},$$

a *binary relation* R between pairs (x_i, x_j) may exist. In which case, we write

$$x_i R x_j$$

and say that x_i *has relation* R *to* x_j.

Relation R may for instance be "is parallel to," "is orthogonal to," or "is congruent to" in geometry. It may be "is greater than," "is a factor of," "is equal to," and so on, in the case when X consists of numbers. On the other hand, if the set X is composed of people, the relation R may be "is son of," "is spouse of," "is friend of," and so forth. Each of these relations is defined only on pairs of numbers of the set, and this is why the name *binary relation*. Although there are relations other than binary (x_i "is a product of" x_j and x_k, for example, will be a tertiary relation), binary relations are the most important in mathematics, and the word "relation" implies a binary relation.

A digraph is the most natural way of representing a binary relation on a set X. Each $x_i \in X$ is represented by a vertex x_i. If x_i has the specified relation R to x_j, a directed edge is drawn from vertex x_i to x_j, for every pair (x_i, x_j). For example, the digraph in Fig. 9-4 represents the relation "is greater than" on a set consisting of five numbers {3, 4, 7, 5, 8}.

Clearly, every binary relation on a finite set can be represented by a digraph without parallel edges. Conversely, every digraph without parallel edges defines a binary relation on the set of its vertices.

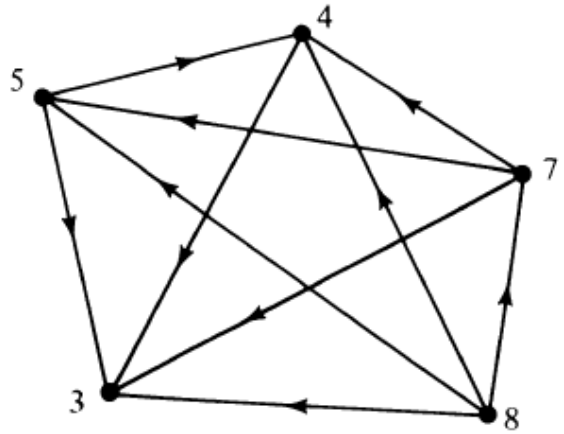


Fig. 9-4 Digraph of a binary relation.

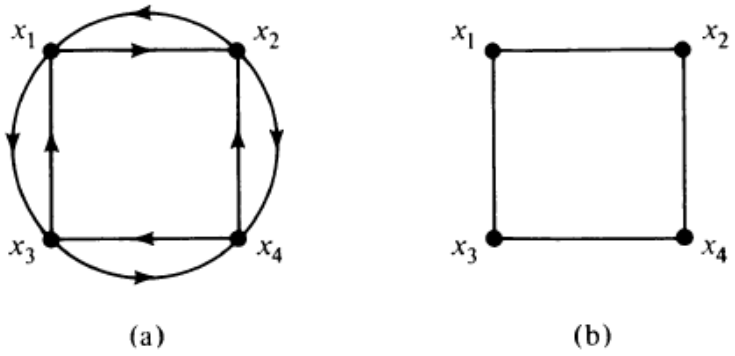


Fig. 9-5 Graphs of symmetric binary relation.

Reflexive Relation: For some relation R it may happen that every element is in relation R to itself. For example, a number is always equal to itself, or a line is always parallel to itself. Such a relation R on set X that satisfies

$$x_i R x_i$$

for every $x_i \in X$ is called a *reflexive* relation. The digraph of a reflexive relation will have a self-loop at every vertex. Such a digraph representing a reflexive binary relation on its vertex set may be called a *reflexive digraph*. A digraph in which no vertex has a self-loop is called an *irreflexive digraph*.

Symmetric Relation: For some relation R it may happen that for all x_i and x_j, if

$$x_i R x_j \text{ holds, then } x_j R x_i \text{ also holds.}$$

Such a relation is called a *symmetric relation*. "Is spouse of" is a symmetric but irreflexive relation. "Is equal to" is both symmetric and reflexive.

The digraph of a symmetric relation is a *symmetric digraph* because for every directed edge from vertex x_i to x_j there is a directed edge from x_j to x_i. Figure 9-5(a) shows the graph of an irreflexive, symmetric binary relation on a set of four elements. The same relation can also be represented by drawing just one undirected edge between every pair of vertices that are related, as in Fig. 9-5(b). Thus every undirected graph is a representation of some symmetric binary relation (on the set of its vertices). Furthermore, every undirected graph with e edges can be thought of as a symmetric digraph with $2e$ directed edges. (A two-way street is equivalent to two one-way streets pointed in opposite directions.)

Transitive Relation: A relation R is said to be *transitive* if for any three elements x_i, x_j, and x_k in the set,

$$x_i R x_j \quad \text{and} \quad x_j R x_k$$

always imply

$$x_i R x_k.$$

The binary relation "is greater than," for example, is a transitive relation. If $x_i > x_j$ and $x_j > x_k$, clearly $x_i > x_k$. "Is descendent of" is another example of a transitive relation.

The digraph of a transitive (but irreflexive and asymmetric) binary relation is shown in Fig. 9-4. Note the triangular subgraphs. A digraph representing a transitive relation (on its vertex set) is called a *transitive directed graph.*

Equivalence Relation: A binary relation is called an *equivalence relation* if it is reflexive, symmetric, and transitive. Some examples of equivalence relations are "is parallel to," "is equal to," "is congruent to," "is equal to modulo m," and "is isomorphic to."

The graph representing an equivalence relation may be called an *equivalence graph.* What does an equivalence graph look like? Let us look at an example, consisting of the equivalence relation "is congruent to modulo 3" defined on the set of 11 integers, 10 through 20. The graph is shown in Fig. 9-6. (Recall that each undirected edge in Fig. 9-6 represents two parallel but oppositely directed edges.)

In Fig. 9-6 we see that the vertex set of the graph is divided into three disjoint classes, each in a separate component. Each component is an undirected subgraph (due to symmetry) with a self-loop at each vertex (due to reflexivity). Furthermore, in each component every vertex is related to (i.e., joined by an edge to) every other vertex.

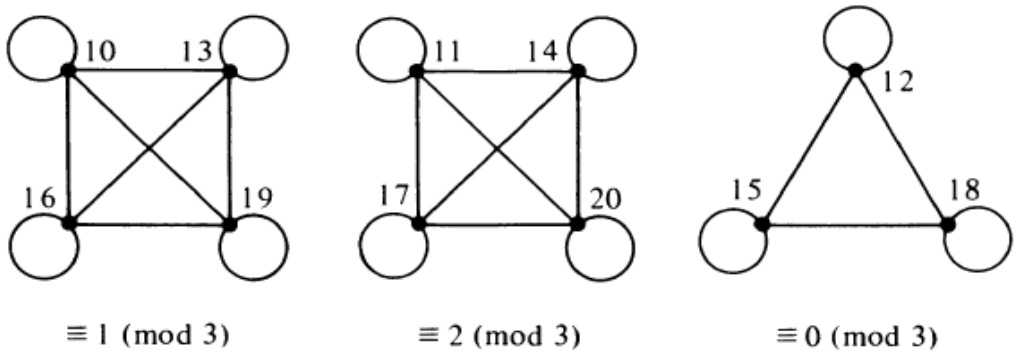


Fig. 9-6 Equivalence graph.

In general, an equivalence relation on a set partitions the elements of the set into classes (called *equivalence classes*) such that two elements are in the same class if and only if they are related. Symmetry ensures that there is no ambiguity regarding membership in the equivalence class; otherwise, x_i may have been related to x_j but not vice versa. Transitivity ensures that in each component every vertex is joined to every other vertex, because if a is related to b and b is related to c, a is also related to c. Transitivity also guarantees that no element can be in more than one class. Reflexivity allows an element to be in a class by itself, if it is not related to any other element in the set.

Relation Matrices: A binary relation R on a set can also be represented by a matrix, called a *relation matrix*. It is a (0, 1), n by n matrix, where n is the number of elements in the set. The i, jth entry in the matrix is 1 if $x_i R x_j$ is true, and is 0, otherwise. For example, the relation matrix of the relation "is greater than" on the set of integers {3, 4, 7, 5, 8} is

$$\begin{array}{c} \\ 3 \\ 4 \\ 7 \\ 5 \\ 8 \end{array} \begin{array}{c} \begin{array}{ccccc} 3 & 4 & 7 & 5 & 8 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix} \end{array}.$$

We shall see in Section 9-8 that this is precisely the adjacency matrix of the digraph representing the binary relation.

DIRECTED PATHS AND CONNECTEDNESS

Walks, paths, and circuits in a directed graph, in addition to being what they are in the corresponding undirected graph, have the added consideration of orientation. For example, in Fig. 9-1, the sequence of vertices and edges $v_5\, e_8\, v_3\, e_6\, v_4\, e_3\, v_1$ is a path "directed" from v_5 to v_1, whereas $v_5\, e_7\, v_4\, e_6\, v_3\, e_1\, v_1$ (although a path in the corresponding undirected graph) has no such consistent direction from v_5 to v_1. A distinction must be made between these two types of paths. It is natural to call the first one a *directed path* from v_5 to v_1, and the second one a *semipath*. The word "path" in a digraph could mean either a directed path or a semipath, and similarly for walks, circuits, and cutsets. More precisely:

A *directed walk* from a vertex v_i to v_j is an alternating sequence of vertices and edges, beginning with v_i and ending with v_j, such that each edge is oriented from the vertex preceding it to the vertex following it. Of course, no edge in a directed walk appears more than once, but a vertex may appear more than once, just as in the case of undirected graphs. A *semiwalk* in a directed graph is a walk in the corresponding undirected graph, but is *not* a *directed walk*. A *walk* in a digraph can mean either a directed walk or a semiwalk.

The definitions of *circuit*, *semicircuit*, and *directed circuit* can be written similarly. Let us turn to Fig. 9-1 once more. The set of edges $\{e_1, e_6, e_3\}$ is a directed circuit. But $\{e_1, e_6, e_2\}$ is a semicircuit. Both of them are circuits.

Connected Digraphs: In Chapter 2 a graph (i.e., undirected graph) was defined as connected if there was at least one path between every pair of vertices. In a digraph there are two different types of paths. Consequently, we have two different types of connectedness in digraphs. A digraph G is said to be *strongly connected* if there is at least one directed path from every vertex to every other vertex. A digraph G is said to be *weakly connected* if its corresponding undirected graph is connected but G is not strongly connected. In Fig. 9-2 one of the digraphs is strongly connected, and the other one is weakly connected. The statement that a digraph G is connected simply means that its corresponding undirected graph is connected; and thus G may be strongly or weakly connected. A directed graph that is not connected is dubbed as disconnected.

Since there are two types of connectedness in a digraph, we can define two types of components also. Each maximal connected (weakly or strongly) subgraph of a digraph G will still be called a *component* of G. But within each component of G the maximal strongly connected subgraphs will be called the *fragments* (or *strongly connected fragments*) of G.

For example, the digraph in Fig. 9-7 consists of two components. The component g_1 contains three fragments $\{e_1, e_2\}$, $\{e_5, e_6, e_7, e_8\}$, and $\{e_{10}\}$. Observe that e_3, e_4, and e_9 do not appear in any fragment of g_1.

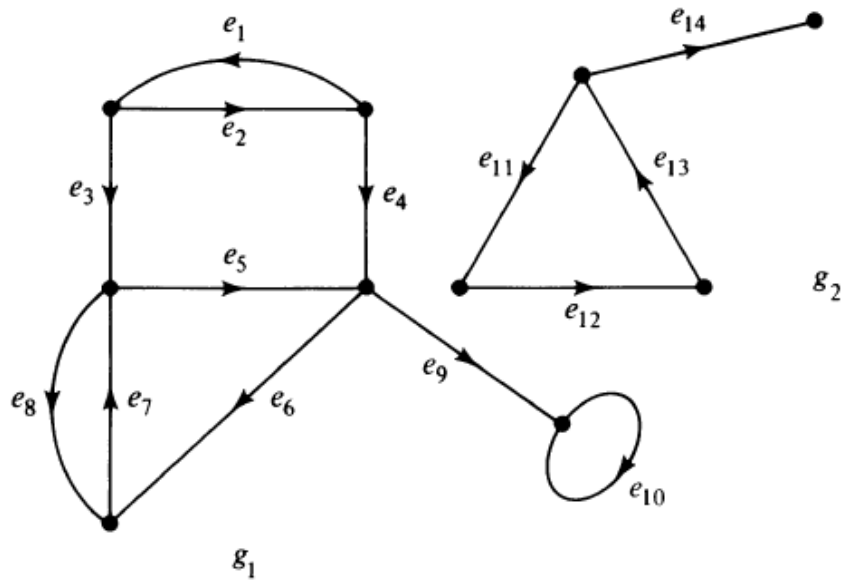


Fig. 9-7 Disconnected digraph with two components.

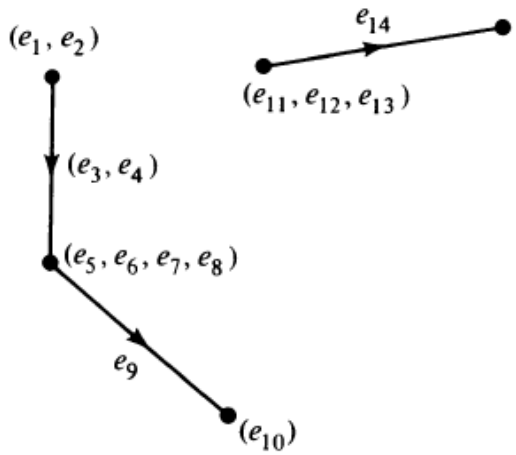


Fig. 9-8 Condensation of Fig. 9-7.

Condensation: The *condensation* G_c of a digraph G is a digraph in which each strongly connected fragment is replaced by a vertex, and all directed edges from one strongly connected component to another are replaced by a single directed edge. The condensation of the digraph G in Fig. 9-7 is shown in Fig. 9-8.

Two observations can be made from the definition:

1. The condensation of a strongly connected digraph is simply a vertex.
2. The condensation of a digraph has no directed circuit.

Accessibility: In a digraph a vertex b is said to be *accessible* (or *reachable*) from vertex a if there is a directed path from a to b. Clearly, a digraph G is strongly connected if and only if every vertex in G is accessible from every other vertex.

TYPES OF ENUMERATION

All graph-enumeration problems fall into two categories:

1. Counting the number of different graphs (or digraphs) of a particular kind, for example, all connected, simple graphs with eight vertices and two circuits.
2. Counting the number of subgraphs of a particular type in a given graph G, such as the number of edge-disjoint paths of length k between vertices a and b in G.

The second type of problem usually involves a matrix representation of graph G and manipulations of this matrix. Such problems, although often encountered in practical applications, are not as varied and interesting as those in the first category. We shall not consider such problems in this chapter.

In problems of type 1 the word "different" is of utmost importance and must be clearly understood. If the graphs are labeled (i.e., each vertex is assigned a name distinct from all others), all graphs are counted. On the other hand, in the case of unlabeled graphs the word "different" means non-isomorphic, and each set of isomorphic graphs is counted as one.

As an example, let us consider the problem of constructing all simple graphs with n vertices and e edges. There are $n(n-1)/2$ unordered pairs of vertices. If we regard the vertices as distinguishable from one another (i.e., labeled graphs), there are

$$\binom{\frac{n(n-1)}{2}}{e} \tag{10-1}$$

ways of selecting e edges to form the graph. Thus expression (10-1) gives the number of simple *labeled* graphs with n vertices and e edges.

Many of these graphs, however, are isomorphic (that is, they are the same except for the labels of their vertices). Hence the number of simple, *unlabeled* graphs of n vertices and e edges is much smaller than that given by (10-1).

Among a collection of graphs, isomorphism is an equivalence relation (Problem 10-1). The number of different unlabeled graphs (of a certain type) equals the number of equivalence classes, under isomorphism, of the labeled graphs. For example, we have 16 different labeled trees of four vertices (Fig. 3-15), and these trees fall into two equivalence classes, under isomorphism. In Fig. 3-15 the 4 trees in the top row fall into one equivalence class, and the remaining 12 into another. Thus we have only two different unlabeled trees of four vertices (Fig. 3-16).

Let us now proceed with counting certain specific types of graphs.

THEOREM 10-1

The number of simple, labeled graphs of n vertices is

$$2^{n(n-1)/2}. \tag{10-2}$$

Proof: The numbers of simple graphs of n vertices and $0, 1, 2, \ldots, n(n-1)/2$ edges are obtained by substituting $0, 1, 2, \ldots, n(n-1)/2$ for e in expression (10-1). The sum of all such numbers is the number of all simple graphs with n vertices. Then the use of the following identity proves the theorem:

$$\binom{k}{0} + \binom{k}{1} + \binom{k}{2} + \cdots + \binom{k}{k-1} + \binom{k}{k} = 2^k. \quad \blacksquare$$

COUNTING LABELED TREES

THEOREM 3-10

There are n^{n-2} labeled trees with n vertices ($n \geq 2$).

Proof of Theorem 3-10: Let the n vertices of a tree T be labeled $1, 2, 3, \ldots, n$. Remove the pendant vertex (and the edge incident on it) having the smallest label, which is, say, a_1. Suppose that b_1 was the vertex adjacent to a_1. Among the remaining $n-1$ vertices let a_2 be the pendant vertex with the smallest label, and b_2 be the vertex adjacent to a_2. Remove the edge (a_2, b_2). This operation is repeated on the remaining $n-2$ vertices, and then on $n-3$ vertices, and so on. The process is terminated after $n-2$ steps, when only two vertices are left. The tree T defines the sequence

$$(b_1, b_2, \ldots, b_{n-2}) \tag{10-3}$$

uniquely. For example, for the tree in Fig. 10-1 the sequence is (1, 1, 3, 5, 5, 5, 9). Note that a vertex i appears in sequence (10-3) if and only if it is not pendant (see Problem 10-2).

Conversely, given a sequence (10-3) of $n - 2$ labels, an n-vertex tree can be

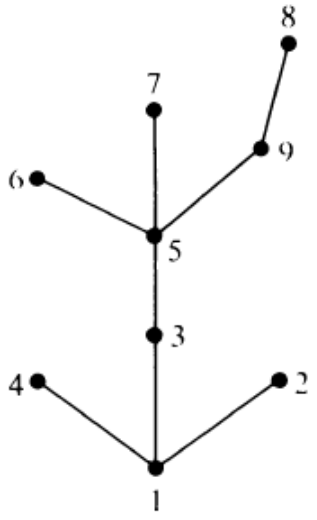


Fig. 10-1 Nine-vertex labeled tree, which yields sequence (1, 1, 3, 5, 5, 5, 9).

constructed uniquely, as follows: Determine the first number in the sequence

$$1, 2, 3, \ldots, n \tag{10-4}$$

that does not appear in sequence (10-3). This number clearly is a_1. And thus the edge (a_1, b_1) is defined. Remove b_1 from sequence (10-3) and a_1 from (10-4). In the remaining sequence of (10-4) find the first number that does not appear in the remainder of (10-3). This would be a_2, and thus the edge (a_2, b_2) is defined. The construction is continued till the sequence (10-3) has no element left. Finally, the last two vertices remaining in (10-4) are joined. For example, given a sequence

$$(4, 4, 3, 1, 1),$$

we can construct a seven-vertex tree as follows: (2, 4) is the first edge. The second is (5, 4). Next, (4, 3). Then (3, 1), (6, 1), and finally (7, 1), as shown in Fig. 10-2.

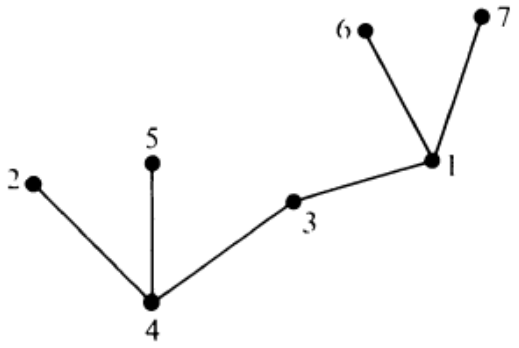


Fig. 10-2 Tree constructed from sequence (4, 4, 3, 1, 1).

For each of the $n - 2$ elements in sequence (10-3) we can choose any one of n numbers, thus forming

$$n^{n-2} \tag{10-5}$$

$(n - 2)$-tuples, each defining a distinct labeled tree of n vertices. And since each tree defines one of these sequences uniquely, there is a one-to-one correspondence between the trees and the n^{n-2} sequences. Hence the theorem. ■

Rooted Labeled Trees: In a rooted graph one vertex is marked as the root. For each of the n^{n-2} labeled trees we have n rooted labeled trees, because any of the n vertices can be made a root. Therefore,

THEOREM 10-2

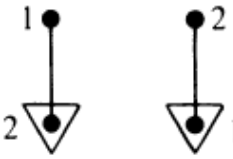
The number of different rooted, labeled trees with n vertices is

$$n^{n-1}. \tag{10-6}$$

All rooted trees for $n = 1$, 2, and 3 are given in Fig. 10-3.

COUNTING UNLABFLED TREES

The problem of enumeration of unlabeled trees is more involved and requires familiarity with the concepts of *generating functions* and *partitions*.

n	Labeled free trees	Labeled rooted trees
1		
2		

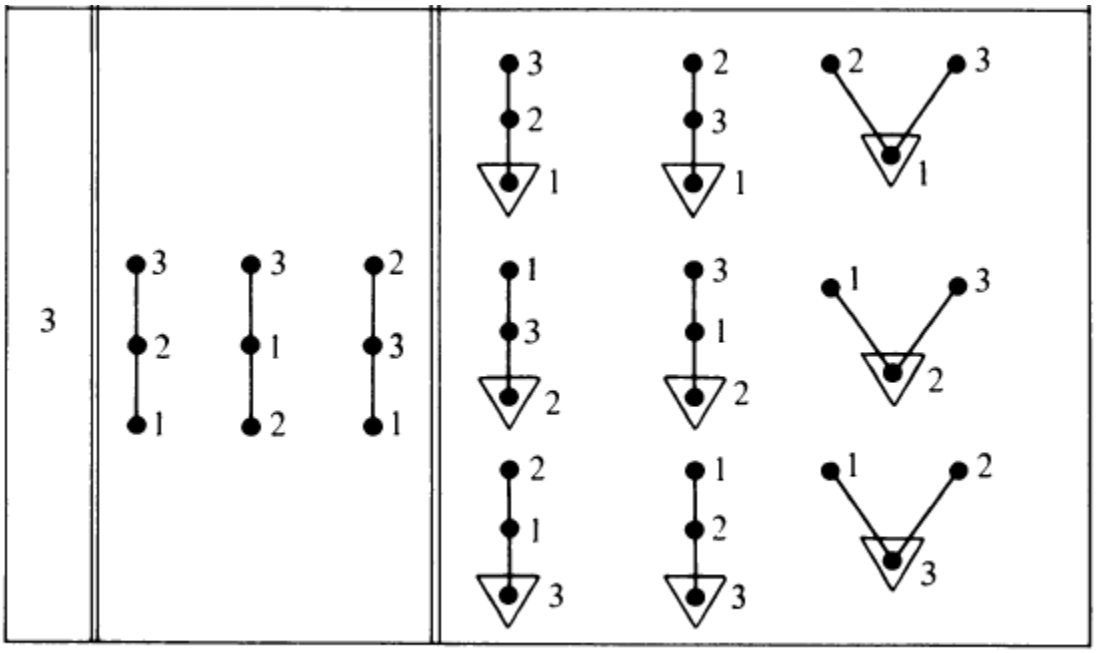


Fig. 10-3 Rooted labeled trees of one, two, and three vertices.

Centroid

In a tree T, at any vertex v of degree d, there are d subtrees with only vertex v in common. The *weight of each subtree* at v is defined as the number of branches in the subtree. Then the *weight of the vertex* v is defined as the weight of the heaviest of the subtrees at v. A vertex with the smallest weight in the entire tree T is called a *centroid* of T.

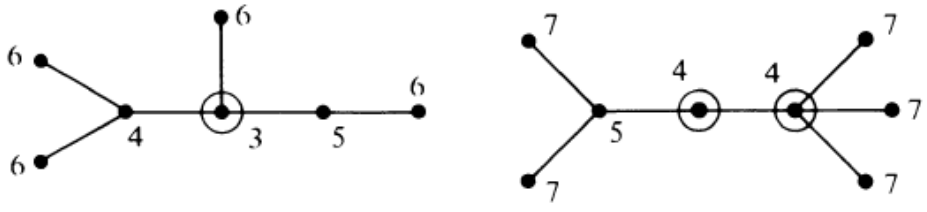
Just as in the case of centers of a tree (Section 3-4), it can be shown that every tree has either one centroid or two centroids. It can also be shown that if a tree has two centroids, the centroids are adjacent. In Fig. 10-6 a tree with a centroid (called a *centroidal tree*) and a tree with two centroids (called a *bicentroidal tree*) are shown. The centroids are shown enclosed in circles, and the numbers next to the vertices are the weights.

Free Unlabeled Trees

Let $t'(x)$ be the counting series for centroidal trees, and $t''(x)$ be the counting series for bicentroidal trees. Then $t(x)$, the counting series for all (unlabeled, free) trees, is the sum of the two. That is,

$$t(x) = t'(x) + t''(x). \tag{10-22}$$

To obtain $t''(x)$, observe that an n-vertex bicentroidal tree can be regarded as consisting of two rooted trees each with $n/2 = m$ vertices, and joined at their roots by an edge. (A bicentroidal tree will always have an even number of vertices; why?) Thus the number of bicentroidal trees with $n = 2m$ vertices is



(a) Centroidal Tree (b) Bicentroidal Tree

given by

$$t_n'' = \binom{u_m + 1}{2} = \frac{u_m(u_m + 1)}{2},$$

and therefore

$$\begin{aligned} t''(x) &= \sum_{m=1}^{\infty} \frac{u_m(u_m + 1)}{2} x^{2m} \\ &= \frac{1}{2} \sum_{m=1}^{\infty} u_m x^{2m} + \frac{1}{2} \sum_{m=1}^{\infty} (u_m x^m)^2 \qquad (10\text{-}23) \\ &= \frac{1}{2} u(x^2) + \frac{1}{2} \sum_{m=1}^{\infty} (u_m x^m)^2. \end{aligned}$$

The number of vertices, n, in a centroidal tree can be odd or even. If n is odd, the maximum weight the centroid could have is $\frac{1}{2}(n - 1)$. This maximum is achieved only when the tree consists of a path of $n - 1$ edges. On the other hand, if n is even and the tree is centroidal, the maximum weight the centroid could possibly have is $\frac{1}{2}(n - 2)$. This maximum is achieved when the degree of the centroid is three, and one of the subtrees consists of just one edge.

Thus, regardless whether n is odd or even, it is clear that an n-vertex (free) centroidal tree can be regarded as composed of several rooted trees, rooted at the centroid, and none of these rooted trees can have more than $\lfloor (n - 1)/2 \rfloor$ edges, where $\lfloor x \rfloor$ denotes the largest integer no greater than x. In view of this observation, an involved manipulation of Eq. (10-21) leads to the following (for missing steps see [10-3]):

$$t'(x) = u(x) - \frac{1}{2}u^2(x) - \frac{1}{2}\sum_{m=1}^{\infty}(u_m x^m)^2. \qquad (10\text{-}24)$$

Adding (10-23) and (10-24), we get the desired counting series:

$$t(x) = u(x) - \frac{1}{2}\Big(u^2(x) - u(x^2)\Big). \qquad (10\text{-}25)$$

This relation, which gives the tree-counting series in terms of the rooted-tree counting series, was first obtained by Richard Otter in 1948 and is known as Otter's formula. The first 10 terms of (10-25) are

$$t(x) = x + x^2 + x^3 + 2x^4 + 3x^5 + 6x^6 + 11x^7 + 23x^8 + 47x^9 + 106x^{10} + \cdots.$$

The reader is encouraged to extend it by another 10 terms. The first 26 terms of both $u(x)$ and $t(x)$ are given in Riordan's book [3-11], page 138.

By now you must have the impression that enumeration of graphs is an involved subject. And indeed it is. So far we have enumerated only four types of graphs—rooted and free trees, both labeled and unlabeled varieties. It is difficult to proceed further without some additional enumerative tool. This is provided by a general counting theorem due to Pólya. We shall first state and discuss Pólya's theorem and then show how it can be applied for counting graphs.

PÓLYA'S COUNTING THEOREM

Permutation

On a finite set A of some objects, a permutation π is a one-to-one mapping from A onto itself. For example, consider a set $\{a, b, c, d\}$. A permutation

$$\pi_1 = \begin{pmatrix} a & b & c & d \\ b & d & c & a \end{pmatrix}$$

takes a into b, b into d, c into c, and d into a. Alternatively, we could write

$$\pi_1(a) = b,$$
$$\pi_1(b) = d,$$

$$\pi_1(c) = c,$$
$$\pi_1(d) = a.$$

The number of elements in the object set on which a permutation acts is called the *degree* of the permutation. The degree of π_1 in the above example is four.

A permutation can also be represented by a digraph, in which each vertex represents an element of the object set and the directed edges represent the mapping. For example, the permutation $\pi_1 = \begin{pmatrix} a & b & c & d \\ b & d & c & a \end{pmatrix}$ is represented diagrammatically by Fig. 10-7.

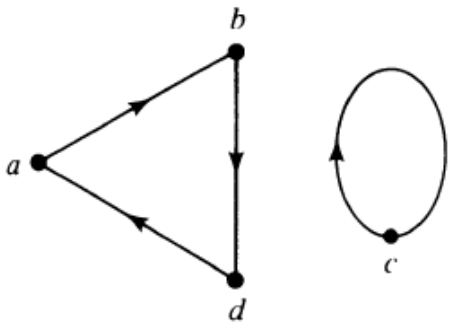


Fig. 10-7 Digraph of a permutation.

Observe that the in-degree and the out-degree of every vertex in the digraph of a permutation is one. Such a digraph must decompose into one or more vertex-disjoint directed circuits (why?). This suggests yet another way of representing a permutation—as a collection of the vertex-disjoint, directed circuits (called the *cycles of the permutation*). Permutation $\begin{pmatrix} a & b & c & d \\ b & d & c & a \end{pmatrix}$ can thus be written as $(a\ b\ d)(c)$. This compact and popular representation is called the *cyclic representation* of a permutation. The number of edges in a permutation cycle is called the *length of the cycle in the permutation.*

Often the only information of interest about a permutation is the number of cycles of various lengths. A permutation π of degree k is said to be of *type* $(\sigma_1, \sigma_2, \ldots, \sigma_k)$ if π has σ_i cycles of length i for $i = 1, 2, \ldots, k$. For example, permutation $(a\ b\ d)(c)$ is of type $(1, 0, 1, 0)$ and permutation $(a\ b\ f)(c)(d\ e\ h)(g)$ is of type $(2, 0, 2, 0, 0, 0, 0, 0)$. Clearly,

$$1\sigma_1 + 2\sigma_2 + 3\sigma_3 + \cdots + k\sigma_k = k. \tag{10-26}$$

Another useful method for indicating the type of a permutation is to introduce k dummy variables, say, $y_1, y_2, \ldots, y_k$, and then show the type of permutation by the expression

$$y_1^{\sigma_1} y_2^{\sigma_2} \cdots y_k^{\sigma_k}. \tag{10-27}$$

Expression (10-27) is called the *cycle structure* of π. For example, the cycle structure of the eight-degree permutation $(a\ b\ f)(c)(d\ e\ h)(g)$ is

$$y_1^2 y_2^0 y_3^2 y_4^0 y_5^0 y_6^0 y_7^0 y_8^0 = y_1^2 y_3^2.$$

Note that the dummy variable y_i has no significance except as a symbol to which subscripts (indicating the lengths) and exponents (indicating the number of cycles) are attached. Two distinct permutations (acting on the same object set) may have the same cycle structure (page 149 in [10-1]).

So far we have discussed only the representation and properties of a permutation individually. Let us now examine a set of permutations collectively.

On a set A with k objects, we have a total of $k!$ possible permutations—including the *identity permutation*, which takes every element into itself. For example, the following are the six permutations on a set of three elements $\{a, b, c\}$:

$$(a)(b)(c),\quad (a\ b)(c),\quad (a\ c)(b),\quad (a)(b\ c),\quad (a\ b\ c),\quad (a\ c\ b).$$

Their cycle structures, respectively, are

$$y_1^3,\quad y_1y_2,\quad y_1y_2,\quad y_1y_2,\quad y_3,\quad y_3. \tag{10-28}$$

Composition of Permutations

Consider the two permutations π_1 and π_2 on an object set $\{1, 2, 3, 4, 5\}$:

$$\pi_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 1 & 4 & 5 & 3 \end{pmatrix} \quad \text{and} \quad \pi_2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 4 & 1 & 2 & 5 \end{pmatrix}.$$

A *composition* of these two permutations $\pi_2\pi_1$ is another permutation obtained by first applying π_1 and then applying π_2 on the resultant. That is,

$$\pi_2\pi_1(1) = \pi_2(2) = 4,$$
$$\pi_2\pi_1(2) = \pi_2(1) = 3,$$
$$\pi_2\pi_1(3) = \pi_2(4) = 2,$$
$$\pi_2\pi_1(4) = \pi_2(5) = 5,$$
$$\pi_2\pi_1(5) = \pi_2(3) = 1.$$

Thus
$$\pi_2\pi_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 3 & 2 & 5 & 1 \end{pmatrix}.$$

Thus among a collection of permutations on the same object set, composition is a binary operation.

Permutation Group

A collection of m permutations $P = \{\pi_1, \pi_2, \ldots, \pi_m\}$ acting on a set

$$A = \{a_1, a_2, \ldots, a_k\}$$

forms a group under composition, if the four postulates† of a group, that is, closure, associativity, identity, and inverse (see Section 6-1), are satisfied. Such a group is called a *permutation group*. For example, it can be easily verified that the set of four permutations

$$\{(a)(b)(c)(d), \quad (a\ c)(b\ d), \quad (a\ b\ c\ d), \quad (a\ d\ c\ b)\} \tag{10-29}$$

acting on the object set $\{a, b, c, d\}$ forms a permutation group.

The number of permutations m in a permutation group is called its *order*, and the number of elements in the object set on which the permutations are acting is called the *degree of the permutation group*. In the example just cited, both the degree and order of the permutation group is four. It can be shown that the set of all $k!$ permutations on a set A of k elements forms a permutation group. Such a group, of order $k!$ and degree k, is called the *full symmetric group*, S_k.

Pólya's Counting Theorem

Let us consider two finite sets, domain D and range R, together with a permutation group P on D. To each element $\rho \in R$ let us assign a quantity $w[\rho]$ and call it the *content* (or *weight*) of the element ρ. The weight $w[\rho]$ can be a symbol or a real number. A mapping f from D to R can be described by a sequence of $|D|$ elements of set R such that the ith element in the sequence is the image of the ith element of set D under f. Therefore the content $W(f)$ of a mapping f can be defined as the product of the contents of all its images. That is,

$$W(f) = \prod_{d \in D} w[f(d)].$$

Clearly, all functions belonging to the same equivalence class defined by (10-33) have identical weights. Therefore, we define the weight of an entire equivalence class (of functions from domain D to range R) to be the (common) weight of the functions in this class. Our problem is to count the number of equivalence classes with various weights, given D, R, permutation group P on D, and weights $w[\rho]$ for each $\rho \in R$. This is exactly what Pólya's counting theorem gives.

In Pólya's terminology, elements ρ of set R are called *figures*, and functions f from D to R are called *configurations*. Often the weights of the elements of R can be expressed as powers of some common quantity x. In that case the weight assignment to elements of set R can be neatly described by means of a counting series $A(x)$

$$A(x) = \sum_{q=0}^{\infty} a_q x^q, \tag{10-34}$$

where a_q is the number of elements in set R with weight x^q.† Likewise, the number of configurations can be expressed in terms of another series, called *configuration counting series* $B(x)$, such that

$$B(x) = \sum_{m=0}^{\infty} b_m x^m, \tag{10-35}$$

where b_m is the number of different configurations having weight x^m. Now we can state the following powerful result known as Pólya's counting theorem.

THEOREM 10-3

The configuration-counting series $B(x)$ is obtained by substituting the figure-counting series $A(x^i)$ for each y_i in the cycle index $Z(P; y_1, y_2, \ldots, y_k)$ of the permutation group P. That is,

$$B(x) = Z(P; \sum a_q x^q, \sum a_q x^{2q}, \sum a_q x^{3q}, \ldots, \sum a_q x^{kq}). \qquad (10\text{-}36)$$

The proof of Pólya's theorem, although not complicated, is not particularly illuminating and is therefore left out. The reader can find it in [10-1], page 157. Our interest is mainly in the application of the theorem; let us illustrate it with some examples.

Example 1: Suppose that we are given a cube and four (identical) balls. In how many ways can the balls be arranged on the corners of the cube? Two arrangements are considered the same if by any rotation of the cube they can be transformed into each other.

The answer is seven, as can be seen by inspection in Fig. 10-9. In Pólya's terms the domain D is the set of the eight corners of the cube, and the range

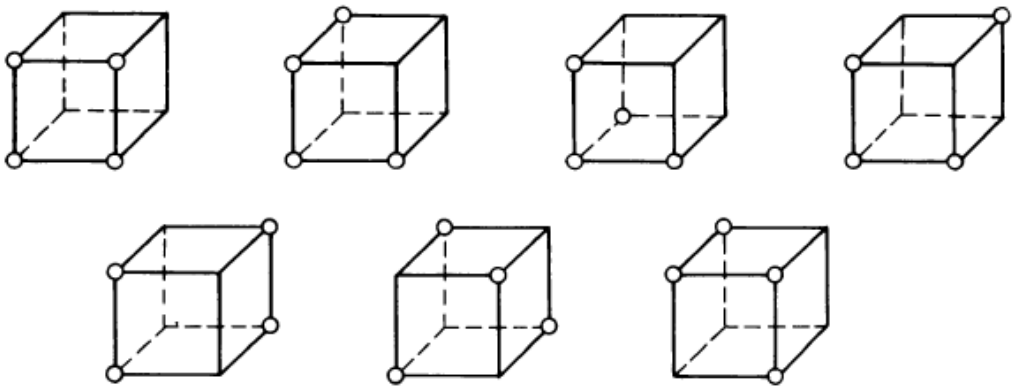


Fig. 10-9 Attaching four balls to corners of a cube.

R consists of two elements (i.e., figures), "presence of a ball" or "absence of a ball," with contents x^1 and x^0, respectively. The figure-counting series is

$$A(x) = \sum_{q=0}^{\infty} a_q x^q = a_0 x^0 + a_1 x^1 = 1 + x, \qquad (10\text{-}37)$$

since a_0, the number of figures with content 0, is one, and a_1, the number of figures with content 1, is also one. The configurations are $2^8 = 256$ different mappings that assign balls to the corners of the cube. The permutation group P on D is the set of all those permutations that can be produced by rotations of the cube. These permutations with their cycle structures are

1. One identity permutation. Its cycle structure is y_1^8.
2. Three 180° rotations around lines connecting the centers of opposite faces. Its cycle structure is y_2^4.
3. Six 90° rotations (clockwise and counterclockwise) around lines connecting the centers of opposite faces. The cycle structure is y_4^2.
4. Six 180° rotations around lines connecting the midpoints of opposite edges. The corresponding cycle structure is y_2^4.
5. Eight 120° rotations around lines connecting opposite corners in the cube. The cycle structure of the corresponding permutation is $y_1^2 y_3^3$.

The cycle index of this group consisting of these 24 permutations is, therefore,

$$Z(P) = \frac{1}{24}(y_1^8 + 9y_2^4 + 6y_4^2 + 8y_1^2 y_3^2). \tag{10-38}$$

Using Pólya's theorem, we now substitute the figure-counting series, that is $1 + x$ for y_1, $1 + x^2$ for y_2, $1 + x^3$ for y_3, and $1 + x^4$ for y_4. This yields the configuration-counting series.

$$B(x) = 1 + x + 3x^2 + 3x^3 + 7x^4 + 3x^5 + 3x^6 + x^7 + x^8. \tag{10-39}$$

The coefficient of x^4 in $B(x)$ gives the number of P-inequivalent configurations of content x^4 (i.e., with four balls). This verifies the answer obtained by exhaustive inspection in Fig. 10-9.

The total number of P-inequivalent configurations (with contents $x^0, x^1, x^2, \ldots, x^8$) is obtained by adding all coefficients in (10-39), which is 23. It may be observed that this is the number of distinct ways of painting

the eight vertices of a cube with two colors (one color corresponds to the "presence of a ball" and the other with the "absence of a ball").

Example 2: In example 1 we were given four identical balls. Now suppose that we are given two red balls and two blue balls, and are again asked to find the number of distinct arrangements on the corners of the cube. Clearly, D, P, and $Z(P)$ will remain the same as they were in example 1. Only the range R and the figure-counting series $A(x)$ will change. The range will contain three elements: (1) presence of no ball, (2) presence of a red ball, and (3) presence of a blue ball. Choosing x to indicate the presence of a red ball and x' to indicate the presence of a blue ball, the three elements in the range mentioned above will have the contents $x^0x'^0$, $x^1x'^0$, and $x^0x'^1$, respectively. Therefore the figure-counting series is

$$A(x, x') = x^0x'^0 + x^1x'^0 + x^0x'^1 = 1 + x + x'.$$

Substituting this figure-counting series in (10-38), we get the configuration-counting series

$$\begin{aligned} B(x, x') = \frac{1}{24}[&(1 + x + x')^8 + 9(1 + x^2 + x'^2)^4 + 6(1 + x^4 + x'^4)^2 \\ &+ 8(1 + x + x')^2(1 + x^3 + x'^3)^2] \\ = 1 &+ x + x' + 3x^2 + 3x'^2 + 3xx' + 3x^3 + 3x'^3 \\ &+ 7x^2x' + 7xx'^2 + 7x^4 + 7x'^4 + 13x^3x' + 13xx'^3 \\ &+ 22x^2x'^2 + 3x^5 + 3x'^5 + 13x^4x' + 13xx'^4 \\ &+ 24x^3x'^2 + 24x^2x'^3 + 3x^6 + 3x'^6 + 7x^5x' \\ &+ 7xx'^5 + 22x^4x'^2 + 22x^2x'^4 + 24x^3x'^3 + x^7 + x'^7 \\ &+ 3x^6x' + 3xx'^6 + 7x^5x'^2 + 7x^2x'^5 + 13x^3x'^4 + 13x^4x'^3 \\ &+ x^8 + x'^8 + x^7x' + xx'^7 + 3x^6x'^2 + 3x^2x'^6 + 3x^5x'^3 \\ &+ 3x^3x'^5 + 7x^4x'^4. \end{aligned} \tag{10-40}$$

The coefficient of $x^rx'^b$ in (10-40) is the number of distinct arrangements with r red balls, b blue balls and $8 - r - b$ corners with no balls. The number of arrangements with two red and two blue balls is, therefore, 22.

For some other non-graph-theoretic examples of the applications of Pólya's theorem, the reader should work out Problems 10-10, 10-11, 10-14, and 10-15. Let us now return to the counting of graphs.

GRAPH ENUMERATION WITH PÓLYA'S THEOREM

Enumeration of Simple Graphs: Let us consider the problem of counting all unlabeled, simple graphs of n vertices. Any such graph G can be regarded as a mapping (i.e., configuration) of the set D of all $\frac{1}{2}n(n-1)$ unordered pairs of vertices (for digraphs $n(n-1)$ pairs of vertices). Range R consists of two elements s and t, with contents x^1 and x^0, respectively. If a vertex pair is joined by an edge in G, the vertex pair maps into s, an element with content x^1; otherwise, into t, an element with content $x^0 = 1$. Thus the figure-counting series is

$$A(x) = \sum a_q x^q = 1 + x.$$

The relevant permutation group in this case is R_n, the group of permutations on the pairs of vertices induced by S_n (the full symmetric group on the n vertices of the graph).† Therefore, the configuration-counting series is obtained by substituting $1 + x$ for y_1, $1 + x^2$ for y_2, $1 + x^3$ for y_3, and so on in $Z(R_n)$. Some specific cases are

(1) For $n = 3$,

$$Z(R_3) = \frac{1}{6}(y_1^3 + 3y_1y_2 + 2y_3).$$

Therefore, the configuration-counting series is

$$B(x) = \frac{1}{6}[(1 + x)^3 + 3(1 + x)(1 + x^2) + 2(1 + x^3)]$$

$$= 1 + x + x^2 + x^3.$$

The coefficient of x^i in $B(x)$ is the number of configurations with content x^i. The content of a configuration here is the number of edges in the corresponding graph. Thus the number of nonisomorphic simple graphs of three vertices with 0, 1, 2, and 3 edges is each one. This is how it should be, as shown in Fig. 10-10.

(2) For $n = 4$, the cycle index $Z(R_4)$ is given in (10-32). Substituting $1 + x^i$ for y_i in (10-32), we get

$$\begin{aligned} B(x) &= \frac{1}{24}[(1 + x)^6 + 9(1 + x)^2(1 + x^2)^2 + 8(1 + x^3)^2 \\ &\quad + 6(1 + x^2)(1 + x^4)] \\ &= 1 + x + 2x^2 + 3x^3 + 2x^4 + x^5 + x^6. \end{aligned} \tag{10-41}$$

In (10-41) the coefficient of x^r gives the number of simple graphs with four vertices and r edges. The validity of series (10-41) is verified in Fig. 10-11.

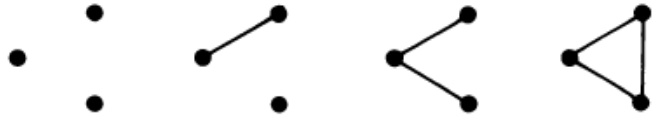


Fig. 10-10 Simple unlabeled graphs of three vertices.

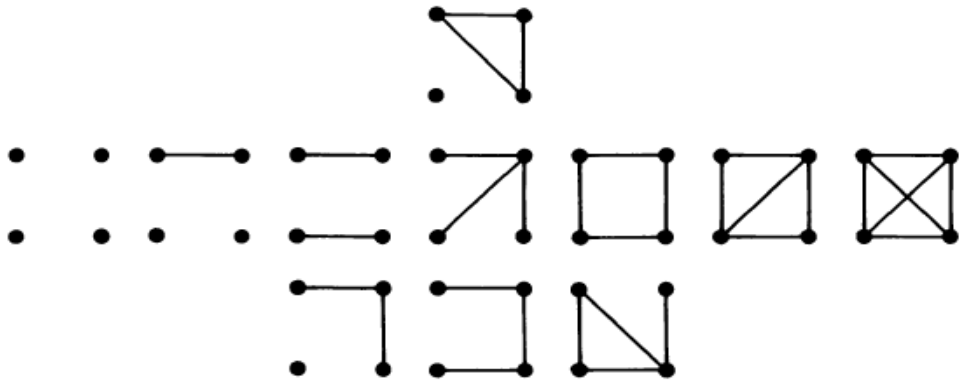


Fig. 10-11 Simple unlabeled graphs of four vertices.

(3) For $n = 5$, the cycle index $Z(R_5)$ is given in Problem 10-9. Substituting $1 + x^i$ for y_i in $Z(R_5)$, we get the counting series $B(x)$ for simple graphs of five vertices, as follows:

$$\begin{aligned} B(x) &= \frac{1}{120}[(1 + x)^{10} + 10(1 + x)^4(1 + x^2)^3 + 20(1 + x)(1 + x^3)^3 \\ &\quad + 15(1 + x)^2(1 + x^2)^4 + 30(1 + x^2)(1 + x^4)^2 \\ &\quad + 20(1 + x)(1 + x^3)(1 + x^6) + 24(1 + x^5)^2] \\ &= 1 + x + 2x^2 + 4x^3 + 6x^4 + 6x^5 + 6x^6 + 4x^7 + 2x^8 \\ &\quad + x^9 + x^{10}. \end{aligned} \tag{10-42}$$

Again, for each r the coefficient of x^r in (10-42) gives the number of simple graphs of five vertices and r edges.

The number of simple, unlabeled graphs with n vertices for any n can be counted similarly.

Enumeration of Multigraphs: Suppose that we are interested in counting multigraphs of n vertices, in which at most two edges are allowed between a pair of vertices.

In this case the domain and the permutation group are the same as they were for simple graphs. The range, however, is different. A pair of vertices may be joined by (1) no edge, (2) one edge, or (3) two edges. Thus range R contains three elements, say, s, t, u, with contents x^0, x^1, and x^2, respectively; that is, x^i indicates the presence of i edges between a vertex pair, for $i = 0, 1, 2$. Threfore, the figure-counting series becomes

$$1 + x + x^2. \tag{10-43}$$

Substitution of $1 + x^r + x^{2r}$ for y_r in $Z(R_n)$ will yield the desired configuration-counting series. For $n = 4$, using the cycle index from (10-32), we get

$$\begin{aligned}&\frac{1}{24}[(1 + x + x^2)^6 + 9(1 + x + x^2)^2(1 + x^2 + x^4)^2 + 8(1 + x^3 + x^6)^2 \\ &\quad + 6(1 + x^2 + x^4)(1 + x^4 + x^8)] \\ &= 1 + x + 3x^2 + 5x^3 + 8x^4 + 9x^5 + 12x^6 + 9x^7 + 8x^8 \\ &\quad + 5x^9 + 3x^{10} + x^{11} + x^{12}.\end{aligned} \tag{10-44}$$

The coefficient of x^i in (10-44) is the number of distinct, unlabeled, multigraphs of four vertices and i edges (such that there are at most two parallel edges between any vertex pair). For example. the coefficient of x^3 is 5, and these five multigraphs are shown in Fig. 10-12.

Instead of allowing at most two parallel edges between a pair of vertices,

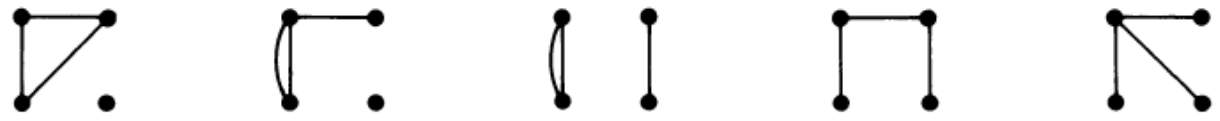


Fig. 10-12 Unlabeled multigraphs of four vertices, three edges, and at most two parallel edges.

had we allowed any number of parallel edges the figure-counting series would be the infinite series

$$A(x) = 1 + x + x^2 + x^3 + \cdots = \frac{1}{1 - x}. \tag{10-45}$$

Enumeration of Digraphs: For enumerating digraphs we have to consider all $n(n-1)$ *ordered* pairs of vertices as constituting the domain. The relevant permutation group will consist of permutations induced on all ordered pairs of vertices by S_n. The cycle index of this permutation group, M_n, can be obtained in the same fashion as was done in the case of R_n. For example, for $n = 4$, Table 10-4 gives the terms in $Z(M_n)$ induced by each term in $Z(S_n)$.

Term in $Z(S_4)$	Induced Term in $Z(M_4)$
y_1^4	y_1^{12}
$y_1^2 y_2$	$y_1^2 y_2^5$
$y_1 y_3$	y_3^4
y_2^2	y_2^6
y_4	y_4^3

Table 10-4

Therefore, the cycle index is

$$Z(M_4) = \frac{1}{24}(y_1^{12} + 6y_1^2 y_2^5 + 8y_3^4 + 3y_2^6 + 6y_4^3). \tag{10-46}$$

For a simple digraph the figure-counting series $A(x) = 1 + x$ is applicable, because a given ordered pair of vertices (a, b) either does or does not have an edge (directed) from a to b. On substituting $1 + x^i$ for every y_i in (10-46), we get the following configuration-counting series for four-vertex, simple digraphs.

$$\begin{aligned} B(x) &= \frac{1}{24}[(1 + x)^{12} + 6(1 + x)^2(1 + x^2)^5 + 8(1 + x^3)^4 \\ &\quad + 3(1 + x^2)^6 + 6(1 + x^4)^3] \\ &= 1 + x + 5x^2 + 13x^3 + 27x^4 + 38x^5 + 48x^6 \\ &\quad + 38x^7 + 27x^8 + 13x^9 + 5x^{10} + x^{11} + x^{12}. \end{aligned} \tag{10-47}$$

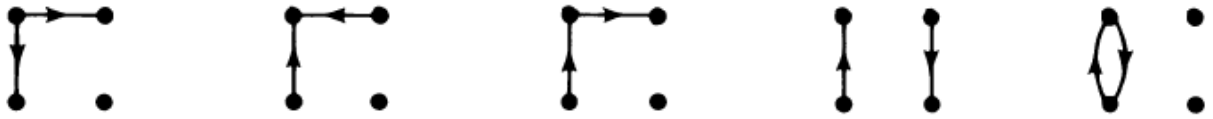


Fig. 10-13 Simple unlabeled digraphs of four vertices and two edges.

The coefficient of x^j in (10-47) is the number of simple digraphs with four vertices and j edges. For example, the five digraphs of two edges are shown in Fig. 10-13.

The general expression for the cycle index, $Z(M_n)$, of the permutation group on $n(n-1)$ ordered pairs induced by S_n is given in [1-5], page 180. Digraphs with parallel edges can be enumerated by substituting the appropriate figure-counting series, say (10-43), in $Z(M_n)$.

UNIT-V

SYLLABUS

DOMINATION IN GRAPHS

Introduction – Terminology and concepts – Applications – Dominating set and domination number – Independent set and independence number – History of domination in graphs.

Introduction

Let p and q be distinct odd primes, so that both of the Legendre symbols (p/q) and (q/p) are defined. It is natural to inquire whether the value of (p/q) can be determined if that of (q/p) is known. To put the question more generally, is there any connection at all between the values of these two symbols? The basic relationship was conjectured experimentally by Euler in 1783 and imperfectly proved by Legendre two years thereafter. Using his symbol, Legendre stated this relationship in the elegant form that has since become known as the Quadratic Reciprocity Law:

$$(p/q)(q/p) = (-1)^{\frac{p-1}{2}\frac{q-1}{2}}.$$

Legendre went amiss in assuming a result which is as difficult to prove as the law itself, namely, that for any prime $p \equiv 1 \pmod 8$, there exists another prime $q \equiv 3 \pmod 4$ for which p is a quadratic residue. Undaunted, he attempted another proof in his *Essai sur la Théorie des Nombres* (1798); this one too contained a gap, since Legendre took for granted that there are an infinite number of primes in certain arithmetical progressions (a fact eventually proved by Dirichlet in 1837, using in the process very subtle arguments from complex variable theory).

At the age of eighteen, Gauss (in 1795), apparently unaware of the work of either Euler or Legendre, rediscovered this reciprocity law and, after a year's unremitting labor, obtained the first complete proof. "It tortured me," says Gauss, "for the whole year and eluded my most strenuous efforts before, finally, I got the proof explained in the fourth section of the *Disquisitiones Arithmeticae*." In the *Disquisitiones Arithmeticae*—which was published in 1801, although finished in 1798—Gauss attributed the Quadratic Reciprocity Law to himself, taking the view that a theorem belongs to the one who gives the first rigorous demonstration. The indignant Legendre was led to complain: "This excessive impudence is unbelievable in a man who has sufficient personal merit not to have the need of appropriating the discoveries of others." All

discussion of priority between the two was futile; since each clung to the correctness of his position, neither took heed of the other. Gauss went on to publish five different demonstrations of what he called "the gem of higher arithmetic," while another was found among his papers. The version presented below, a variant of one of Gauss' own arguments, is due to his student, Ferdinand Eisenstein (1823–1852). The proof is complicated (and it would perhaps be unreasonable to expect an easy proof), but the underlying idea is simple enough.

THEOREM 9-9 (Gauss' Quadratic Reciprocity Law). *If p and q are distinct odd primes, then*

$$(p/q)(q/p) = (-1)^{\frac{p-1}{2}\frac{q-1}{2}}.$$

Proof: Consider the rectangle in the xy coordinate plane whose vertices are $(0, 0)$, $(p/2, 0)$, $(0, q/2)$, and $(p/2, q/2)$. Let R denote the region within this rectangle, not including any of the bounding lines. The general plan of attack is to count the number of lattice points (that is, the points whose coordinates are integers) inside R in two different ways. Since p and q are both odd, the lattice points in R consist of all points (n, m), where $1 \leq n \leq (p-1)/2$ and $1 \leq m \leq (q-1)/2$; the number of such points is clearly

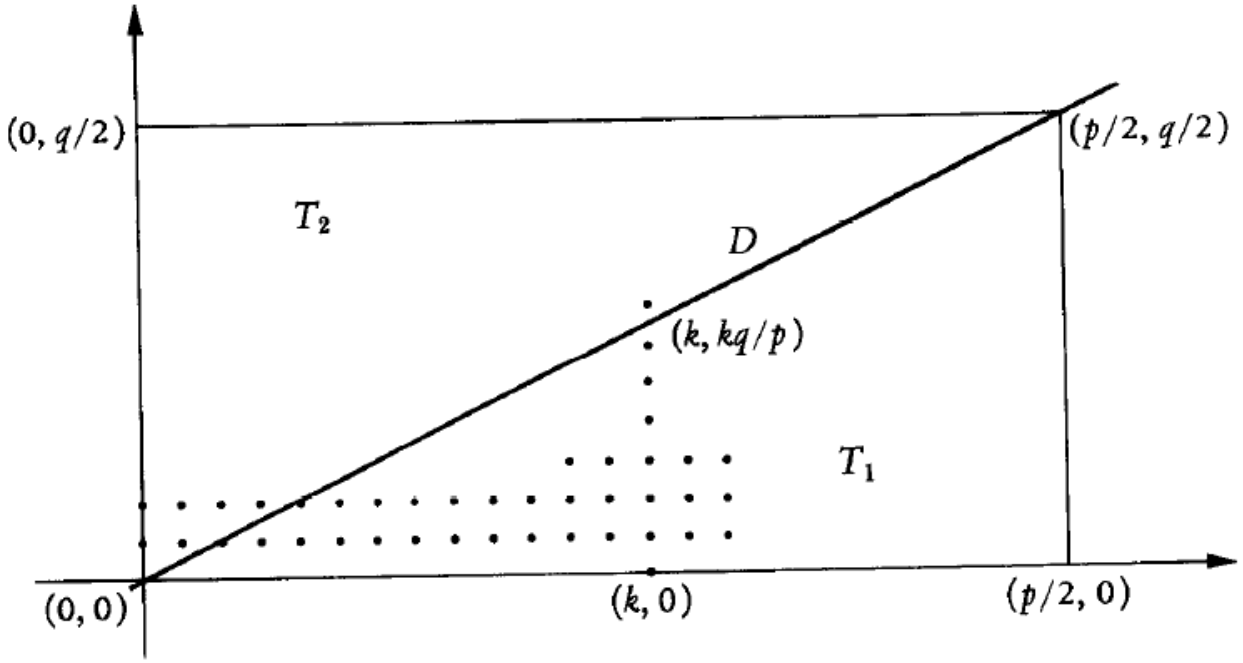
$$\frac{p-1}{2} \cdot \frac{q-1}{2}.$$

Now the diagonal D from $(0, 0)$ to $(p/2, q/2)$ has the equation $y = (q/p)x$, or equivalently, $py = qx$. Since $\gcd(p, q) = 1$, none of the lattice points inside R will lie on D. For p must divide the x coordinate of any lattice point on the line $py = qx$, and q must divide its y coordinate; there are are no such points in R. Suppose that T_1

denotes the portion of R which is below the diagonal D, and T_2 the portion above. By what we have just seen, it suffices to count the lattice points inside each of these triangles.

The number of integers in the interval $0 < y < kq/p$ is $[kq/p]$. Thus, for $1 \leq k \leq (p-1)/2$, there are precisely $[kq/p]$ lattice points in T_1 directly above the point $(k, 0)$ and below D; in other words, lying on the vertical line segment from $(k, 0)$ to $(k, kq/p)$. It follows that the total number of lattice points contained in T_1 is

$$\sum_{k=1}^{(p-1)/2} [kq/p].$$



A similar calculation, with the roles of p and q interchanged, shows that the number of lattice points within T_2 is

$$\sum_{j=1}^{(q-1)/2} [jp/q].$$

This accounts for all of the lattice points inside R, so that

$$\frac{p-1}{2}\cdot\frac{q-1}{2}=\sum_{k=1}^{(p-1)/2}[kq/p]+\sum_{j=1}^{(q-1)/2}[jp/q].$$

The time has come for Gauss' Lemma to do its duty:

$$\begin{aligned}(p/q)(q/p)&=(-1)^{\sum_{j=1}^{(q-1)/2}[jp/q]}\cdot(-1)^{\sum_{k=1}^{(p-1)/2}[kq/p]}\\&=(-1)^{\sum_{j=1}^{(q-1)/2}[jp/q]+\sum_{k=1}^{(p-1)/2}[kq/p]}\\&=(-1)^{\frac{p-1}{2}\frac{q-1}{2}}\end{aligned}$$

The proof of the Quadratic Reciprocity Law is now complete.

COROLLARY 1. *If p and q are distinct odd primes, then*

$$(p/q)(q/p)=\begin{cases}1 & \text{if } p\equiv 1 \pmod 4 \text{ or } q\equiv 1 \pmod 4\\ -1 & \text{if } p\equiv q\equiv 3 \pmod 4\end{cases}$$

Proof: The number $(p-1)/2\cdot(q-1)/2$ is even if and only if at least one of the integers p and q is of the form $4k+1$; if both are of the form $4k+3$, then $(p-1)/2\cdot(q-1)/2$ is odd.

Multiplying each side of the Quadratic Reciprocity equation by (q/p) and using the fact that $(q/p)^2=1$, we could also formulate this as

COROLLARY 2. *If p and q are distinct odd primes, then*

$$(p/q)=\begin{cases}(q/p) & \text{if } p\equiv 1 \pmod 4 \text{ or } q\equiv 1 \pmod 4\\ -(q/p) & \text{if } p\equiv q\equiv 3 \pmod 4\end{cases}$$

Let us see what this last series of results accomplishes. Take p to be an odd prime and $a \neq \pm 1$ to be an integer not divisible by p. Suppose further that a has the factorization

$$a = \pm 2^{k_0} p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r},$$

where the p_i are odd primes. Since the Legendre symbol is multiplicative,

$$(a/p) = (\pm 1/p)(2/p)^{k_0}(p_1/p)^{k_1} \cdots (p_r/p)^{k_r}.$$

In order to evaluate (a/p), we have only to calculate the symbols $(-1/p)$, $(2/p)$, and (p_i/p). The values of $(-1/p)$ and $(2/p)$ were discussed earlier, so that the one stumbling block is (p_i/p), where p_i and p are distinct odd primes; this is where the Quadratic Reciprocity Law enters. For Corollary 2 allows us to replace (p_i/p) by a new Legendre symbol having a smaller denominator. Through continued inversion and division, the computation can be reduced to that of the known quantities

$$(-1/q), (1/q), \text{ and } (2/q).$$

Example 9-5

Consider the Legendre symbol (29/53), for instance. Since both $29 \equiv 1 \pmod 4$ and $53 \equiv 1 \pmod 4$, we see that

$$\begin{aligned}(29/53) = (53/29) = (24/29) &= (2/29)(3/29)(4/29) \\ &= (2/29)(3/29).\end{aligned}$$

With reference to Theorem 9-6, $(2/29) = -1$, while inverting again,

$$(3/29) = (29/3) = (2/3) = -1,$$

where we used the congruence $29 \equiv 2 \pmod 3$. The net effect is that

$$(29/53) = (2/29)(3/29) = (-1)(-1) = 1.$$

The Quadratic Reciprocity Law provides a very satisfactory answer to the problem of finding all odd primes $p \neq 3$ for which 3 is a quadratic residue. Since $3 \equiv 3 \pmod 4$, Corollary 2 above implies that

$$(3/p) = \begin{cases} (p/3) & \text{if } p \equiv 1 \pmod 4 \\ -(p/3) & \text{if } p \equiv 3 \pmod 4. \end{cases}$$

Now $p \equiv 1 \pmod 3$ or $p \equiv 2 \pmod 3$. By Theorems 9-2 and 9-6,

$$(p/3) = \begin{cases} 1 & \text{if } p \equiv 1 \pmod 3 \\ -1 & \text{if } p \equiv 2 \pmod 3 \end{cases}$$

the implication of which is that $(3/p) = 1$ if and only if

(1) $$p \equiv 1 \pmod 4 \quad \text{and} \quad p \equiv 1 \pmod 3,$$

or

(2) $$p \equiv 3 \pmod 4 \quad \text{and} \quad p \equiv 2 \pmod 3.$$

The restrictions in (1) are equivalent to requiring that $p \equiv 1 \pmod{12}$ while those in (2) are equivalent to $p \equiv 11 \equiv -1 \pmod{12}$.

THEOREM 9-10. *If $p \neq 3$ is an odd prime, then*

$$(3/p) = \begin{cases} 1 & \text{if } p \equiv \pm 1 \pmod{12} \\ -1 & \text{if } p \equiv \pm 5 \pmod{12} \end{cases}$$

Example 9-6

The purpose of this example is to investigate the existence of solutions of the congruence

$$x^2 \equiv 196 \pmod{1357}.$$

Since $1357 = 23 \cdot 59$, the given congruence is solvable if and only if both

$$x^2 \equiv 196 \pmod{23} \quad \text{and} \quad x^2 \equiv 196 \pmod{59}$$

are solvable. Our procedure is to find the values of the Legendre symbols (196/23) and (196/59).

The evaluation of (196/23) requires the use of Theorem 9-10:

$$(196/23) = (12/23) = (3/23) = 1.$$

Thus, the congruence $x^2 \equiv 196 \pmod{23}$ admits a solution. As regards the symbol (196/59), the Quadratic Reciprocity Law enables us to write

$$(196/59) = (19/59) = -(59/19) = -(2/19) = -(-1) = 1.$$

It is therefore possible to solve $x^2 \equiv 196 \pmod{59}$ and, in consequence, the congruence $x^2 \equiv 196 \pmod{1357}$ as well.

Let us turn to a quite different application of these ideas. At an earlier stage, it was observed that if $F_n = 2^{2^n} + 1$, $n > 1$, is a prime, then 2 is not a primitive root of F_n. We now possess the means to show that the integer 3 serves as a primitive root of any prime of this type.

As a step in this direction, note that any F_n is of the form $12k + 5$. A simple induction argument confirms that $4^m \equiv 4 \pmod{12}$ for $m = 1, 2, \ldots$; hence, we must have

$$F_n = 2^{2^n} + 1 = 2^{2m} + 1 = 4^m + 1 \equiv 5 \pmod{12}.$$

If F_n happens to be prime, then Theorem 9-10 permits the conclusion

$$(3/F_n) = -1,$$

or, using Euler's Criterion,

$$3^{\frac{F_n-1}{2}} \equiv -1 \pmod{F_n}.$$

Switching to the phi-function, the last congruence says that

$$3^{\phi(F_n)/2} \equiv -1 \pmod{F_n}.$$

From this, it may be inferred that 3 has order $\phi(F_n)$ modulo F_n, and so 3 is a primitive root of F_n.

PROBLEMS

1. Evaluate the following Legendre symbols:
   (a) (71/73), (b) (−219/383), (c) (461/773), (d) (1234/4567), (e) (3658/12703). [*Hint:* $3658 = 2 \cdot 31 \cdot 59$.]
2. Prove that 3 is a quadratic nonresidue of all primes of the form $2^{2n}+1$, as well as all primes of the form $2^p - 1$, where p is an odd prime. [*Hint:* For all n, $4^n \equiv 4 \pmod{12}$.]
3. Determine whether the following quadratic congruences are solvable:
   (a) $x^2 \equiv 219 \pmod{419}$.
   (b) $3x^2 + 6x + 5 \equiv 0 \pmod{89}$.
   (c) $2x^2 + 5x - 9 \equiv 0 \pmod{101}$.
4. Verify that if p is an odd prime, then

$$(-2/p) = \begin{cases} 1 & \text{if } p \equiv 1 \pmod 8 \text{ or } p \equiv 3 \pmod 8 \\ -1 & \text{if } p \equiv 5 \pmod 8 \text{ or } p \equiv 7 \pmod 8 \end{cases}$$

5. (a) Prove that if $p > 3$ is an odd prime, then

$$(-3/p) = \begin{cases} 1 & \text{if } p \equiv 1 \pmod 6 \\ -1 & \text{if } p \equiv 5 \pmod 6 \end{cases}$$

   (b) Using part (a), show that there are infinitely many primes of the form $6k+1$. [*Hint:* Assume that $p_1, p_2, \ldots, p_r$ are all the primes of the form $6k+1$ and consider the integer $(2p_1p_2\cdots p_r)^2 + 3$.]
6. Use Theorem 9-2 and Problems 4 and 5 to determine which primes can divide each of n^2+1, n^2+2, n^2+3 for some value of n.

So far in the proceedings, quadratic congruences with (odd) prime moduli have been of paramount importance. The remaining theorems broaden the horizon by allowing a composite modulus. To start, let us consider the situation where the modulus is a power of a prime.

THEOREM 9-11. *If p is an odd prime and* $\gcd(a, p) = 1$, *then the congruence*

$$x^2 \equiv a \pmod{p^n}, \qquad n \geq 1$$

has a solution if and only if $(a/p) = 1$.

Proof: As is common with many "if and only if" theorems, one half of the proof is trivial while the other half requires considerable effort: If $x^2 \equiv a \pmod{p^n}$ has a solution, then so does $x^2 \equiv a \pmod{p}$ —in fact, the same solution—whence $(a/p) = 1$.

For the converse, suppose that $(a/p) = 1$. We argue that $x^2 \equiv a \pmod{p^n}$ is solvable by inducting on n. If $n = 1$ there is really nothing to prove; indeed, $(a/p) = 1$ is just another way of saying that $x^2 \equiv a \pmod{p}$ can be solved. Assume that the result holds for $n = k \geq 1$, so that $x^2 \equiv a \pmod{p^k}$ admits a solution x_0. Then

$$x_0^2 = a + bp^k$$

for an appropriate choice of b. In passing from k to $k + 1$, we shall use x_0 and b to write down explicitly a solution to the congruence $x^2 \equiv a \pmod{p^{k+1}}$.

Towards this end, we first solve the linear congruence

$$2x_0 y \equiv -b \pmod{p},$$

obtaining a unique solution y_0 modulo p (this is certainly possible, since $\gcd(2x_0, p) = 1$). Next, consider the integer

$$x_1 = x_0 + y_0 p^k.$$

Upon squaring this integer, we get

$$(x_0 + y_0 p^k)^2 = x_0^2 + 2x_0 y_0 p^k + y_0^2 p^{2k}$$
$$= a + (b + 2x_0 y_0) p^k + y_0^2 p^{2k}.$$

But $p \mid (b + 2x_0 y_0)$, from which it follows that

$$x_1^2 = (x_0 + y_0 p^k)^2 \equiv a \pmod{p^{k+1}}.$$

Thus, the congruence $x^2 \equiv a \pmod{p^n}$ has a solution for $n = k + 1$ and, by induction, for all positive integers n.

Let us run through a specific example in detail. The first step in obtaining a solution of, say, the quadratic congruence

$$x^2 \equiv 23 \pmod{7^2}$$

is to solve $x^2 \equiv 23 \pmod 7$, or what amounts to the same thing, the congruence

$$x^2 \equiv 2 \pmod 7.$$

Since $(2/7) = 1$, a solution surely exists; in fact $x_0 = 3$ is an obvious choice. Now x_0^2 can be represented as

$$3^2 = 9 = 23 + (-2)7,$$

so that $b = -2$ (in our special case, the integer 23 plays the role of a). Following the proof of Theorem 9-11, we next determine y so that

$$6y \equiv 2 \pmod 7;$$

that is, $3y \equiv 1 \pmod 7$. This linear congruence is satisfied by $y_0 = 5$. Hence,

$$x_0 + 7y_0 = 3 + 7 \cdot 5 = 38$$

serves as a solution to the original congruence $x^2 \equiv 23 \pmod{49}$. It should be noted that $-38 \equiv 11 \bmod (49)$ is the only other solution.

If, instead, the congruence

$$x^2 \equiv 23 \pmod{7^3}$$

were proposed for solution, we would start with

$$x^2 \equiv 23 \pmod{7^2},$$

obtaining a solution $x_0 = 38$. Since

$$38^2 = 23 + 29 \cdot 7^2,$$

the integer $b = 29$. We would then find the unique solution $y_0 = 1$ of the linear congruence

$$76y \equiv -29 \pmod 7.$$

Then $x^2 \equiv 23 \pmod{7^3}$ is satisfied by

$$x_0 + y_0 7^2 = 38 + 1 \cdot 49 = 87,$$

as well as $-87 \equiv 256 \pmod{7^3}$.

Having dwelt at length on odd primes, let us now take up the case $p = 2$. The next theorem supplies the pertinent information.

THEOREM 9-12. *Let a be an odd integer. Then*

(1) $x^2 \equiv a \pmod 2$ *always has a solution*;
(2) $x^2 \equiv a \pmod 4$ *has a solution if and only if* $a \equiv 1 \pmod 4$;
(3) $x^2 \equiv a \pmod{2^n}$, *for* $n \geq 3$, *has a solution if and only if* $a \equiv 1 \pmod 8$.

Proof: The first assertion is obvious. The second depends on the observation that the square of any odd integer is congruent to 1 modulo 4. Thus, $x^2 \equiv a \pmod 4$ can be solved only when a is of the form $4k + 1$; in this event, there are two solutions modulo 4, namely $x = 1$ and $x = 3$.

Now consider the case in which $n \geq 3$. Since the square of any odd integer is congruent to 1 modulo 8, we see that for the congruence $x^2 \equiv a \pmod{2^n}$ to be solvable it is necessary that a should be of the form $8k + 1$. To go the other way, let us suppose that $a \equiv 1 \pmod 8$ and proceed by induction on n. When $n = 3$, the congruence $x^2 \equiv a \pmod{2^n}$ is certainly solvable; indeed, each of the integers 1, 3, 5, 7 satisfies $x^2 \equiv 1 \pmod 8$. Fix a value of $n > 3$ and assume, for the induction hypothesis, that the congruence $x^2 \equiv a \pmod{2^n}$ admits a solution x_0. Then there exists an integer b for which

$$x_0^2 = a + b2^n.$$

Since a is odd, so is the integer x_0. It is therefore possible to find a unique solution y_0 of the linear congruence

$$x_0 y \equiv -b \pmod 2.$$

We argue that the integer

$$x_1 = x_0 + y_0 2^{n-1}$$

satisfies the congruence $x^2 \equiv a \pmod{2^{n+1}}$. Squaring yields

$$(x_0 + y_0 2^{n-1})^2 = x_0^2 + x_0 y_0 2^n + y_0^2 2^{2n-2}$$
$$= a + (b + x_0 y_0)2^n + y_0^2 2^{2n-2}.$$

By the way y_0 was chosen, $2 \mid (b + x_0 y_0)$, hence

$$x_1^2 = (x_0 + y_0 2^{n-1})^2 \equiv a \pmod{2^{n+1}}$$

(one also uses the fact that $2n - 2 = n + 1 + (n - 3) > n + 1$). Thus $x^2 \equiv a \pmod{2^{n+1}}$ is solvable, completing the induction step and the proof.

To illustrate: the congruence $x^2 \equiv 5 \pmod 4$ has a solution, but $x^2 \equiv 5 \pmod 8$ does not; on the other hand, $x^2 \equiv 17 \pmod{16}$ and $x^2 \equiv 17 \pmod{32}$ are both solvable.

In theory, we can now completely settle the question of when there exists an integer x such that

$$x^2 \equiv a \pmod n, \quad \gcd(a, n) = 1, \qquad n > 1.$$

For suppose that n has the prime-power decomposition

$$n = 2^{k_0} p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}, \qquad k_0 \geq 0,\ k_i > 0$$

where the p_i are distinct odd primes. Since the problem of solving the quadratic congruence $x^2 \equiv a \pmod n$ is equivalent to that of solving the system of congruences

$$\begin{aligned} x^2 &\equiv a \pmod{2^{k_0}}, \\ x^2 &\equiv a \pmod{p_1^{k_1}}, \\ &\vdots \\ x^2 &\equiv a \pmod{p_r^{k_r}}, \end{aligned}$$

our last two results may be combined to give the following general conclusion.

THEOREM 9-13. *Let* $n = 2^{k_0} p_1^{k_1} \cdots p_r^{k_r}$ *be the prime factorization of* $n > 1$ *and let* $\gcd(a, n) = 1$. *Then* $x^2 \equiv a \pmod n$ *is solvable if and only if*

(1) $(a/p_i) = 1$ *for* $i = 1, 2, \ldots, r$;
(2) $a \equiv 1 \pmod 4$ *if* $4 \mid n$, *but* $8 \nmid n$; $a \equiv 1 \pmod 8$ *if* $8 \mid n$.

PROBLEMS

1. (a) Show that 7 and 18 are the only incongruent solutions of $x \equiv -1 \pmod{5^2}$.
   (b) Use part (a) to find the solutions of $x^2 \equiv -1 \pmod{5^3}$.
2. Solve each of the following quadratic congruences:
   (a) $x^2 \equiv 7 \pmod{3^3}$;
   (b) $x^2 \equiv 14 \pmod{5^3}$;
   (c) $x^2 \equiv 2 \pmod{7^3}$.
3. Solve the congruence $x^2 \equiv 31 \pmod{11^4}$.
4. Find the solutions of $x^2 + 5x + 6 \equiv 0 \pmod{5^3}$ and $x^2 + x + 3 \equiv 0 \pmod{3^3}$.
5. Prove that if the congruence $x^2 \equiv a \pmod{2^n}$, where $n \geq 3$, has a solution, then it has exactly four incongruent solutions. [*Hint:* If x_0 is any solution, then the four integers x_0, $-x_0$, $x_0 + 2^{n-1}$, $-x_0 + 2^{n-1}$ are incongruent modulo 2^n and comprise all the solutions.]
6. From $23^2 \equiv 17 \pmod{2^7}$, find three other solutions of the congruence $x^2 \equiv 17 \pmod{2^7}$.

1 The idea of public key cryptography

Recall that a cryptosystem consists of a 1-to-1 enciphering transformation f from a set $\mathcal{P}$ of all possible plaintext message units to a set $\mathcal{C}$ of all possible ciphertext message units. Actually, the term "cryptosystem" is more often used to refer to a whole family of such transformations, each corresponding to a choice of *parameters* (the sets $\mathcal{P}$ and $\mathcal{C}$, as well as the map f, may depend upon the values of the parameters). For example, for a fixed N-letter alphabet (with numerical equivalents also fixed once and for all), we might consider the affine cryptosystem (or "family of cryptosystems") which for each $a \in (\mathbf{Z}/N\mathbf{Z})^*$ and $b \in \mathbf{Z}/N\mathbf{Z}$ is the map from $\mathcal{P} = \mathbf{Z}/N\mathbf{Z}$ to $\mathcal{C} = \mathbf{Z}/N\mathbf{Z}$ defined by $C \equiv aP + b \ mod \ N$. In this example, the sets $\mathcal{P}$ and $\mathcal{C}$ are fixed (because N is fixed), but the enciphering transformation f depends upon the choice of parameters a, b. The enciphering transformation can then be described by (i) an algorithm, which is the same for the whole family, and (ii) the values of the parameters. The values of the parameters are called the *enciphering key* K_E. In our example, K_E is the pair (a, b). In practice, we shall suppose that the algorithm is publicly known, i.e., the general procedure used to encipher cannot be kept secret. However, the keys can easily be changed periodically and, if one wants, kept secret.

One also needs an algorithm and a key in order to decipher, i.e., compute f^{-1}! The key is called the *deciphering key* K_D. In our example of the affine cryptosystem family, deciphering is also accomplished by an affine map, namely $P \equiv a^{-1}C - a^{-1}b \ mod \ N$, and so the deciphering transformation uses the same algorithm as the enciphering transformation, except

with a different key, namely, the pair $(a^{-1}, -a^{-1}b)$. (In some cryptosystems, the deciphering algorithm, as well as the key, is different from the enciphering algorithm.) We shall always suppose that the deciphering and enciphering algorithms are publicly known, and that it is the keys K_E and K_D which can be concealed.

Let us suppose that someone wishes to communicate secretly using the above affine cryptosystem $C \equiv aP + b$. We saw in § III.1 that it is not hard to break the system if one uses single-letter message units in an N-letter alphabet. It is a little more difficult to break the system if one uses digraphs, which can be regarded as symbols in an N^2-letter alphabet. It would be safer to use blocks of k letters, which have numerical equivalents in $\mathbf{Z}/N^k\mathbf{Z}$. At least for $k > 3$ it is not easy to use frequency analysis, since the number of possible k-letter blocks is very large, and one will find many that are close contenders for the title of most frequently occurring k-graph. If we want to increase k, we must be concerned about the length of time it takes to do various arithmetic tasks (the most important one being finding a^{-1} by the Euclidean algorithm) involved in setting up our keys and carrying out the necessary transformations every time we send a message or our friend at the other end deciphers a message from us. That is, it is useful to have big-O estimates for the order of magnitude of time (as the parameters increase, i.e., as the cryptosystem becomes "larger") that it takes to: encipher (knowing K_E), decipher (knowing K_D), or break the code by enciphering without knowledge of K_E or deciphering without knowledge of K_D.

In all of the examples in Chapter III — and in all of the cryptosystems used historically until about fifteen years ago — it is not really necessary to specify the deciphering key once the enciphering key (and the general algorithms) are known. Even if we are working with large numbers — such as N^k with k fairly large — it is possible to determine the deciphering key from the enciphering key using an order of magnitude of time which is roughly the same as that needed to implement the various algorithms. For example, in the case of an affine enciphering transformation of $\mathbf{Z}/N^k\mathbf{Z}$, once we know the enciphering key $K_E = (a, b)$ we can compute the deciphering key $K_D = (a^{-1} \ mod \ N^k, \ -a^{-1}b \ mod \ N^k)$ by the Euclidean algorithm in $O(log^3(N^k))$ bit operations.

Thus, with a traditional cryptosystem anyone who knew enough to decipher messages could, with little or no extra effort, determine the enciphering key. Indeed, it was considered naive or foolish to think that someone who had broken a cipher might nevertheless not know the enciphering key. We see this in the following passage from the autobiography of a well-known historical personality:

Thus, with a traditional cryptosystem anyone who knew enough to decipher messages could, with little or no extra effort, determine the enciphering key. Indeed, it was considered naive or foolish to think that someone who had broken a cipher might nevertheless not know the enciphering key.

We see this in the following passage from the autobiography of a well-known historical personality:

Thus, with a traditional cryptosystem anyone who knew enough to decipher messages could, with little or no extra effort, determine the enciphering key. Indeed, it was considered naive or foolish to think that someone who had broken a cipher might nevertheless not know the enciphering key.

> Five or six weeks later, she [Madame d'Urfé] asked me if I had deciphered the manuscript which had the transmutation procedure. I told her that I had.

"Without the key, sir, excuse me if I believe the thing impossible."

"Do you wish me to name your key, madame?"

"If you please."

I then told her the key-word, which belonged to no language, and I saw her surprise. She told me that it was impossible, for she believed herself the only possessor of that word which she kept in her memory and which she had never written down.

I could have told her the truth — that the same calculation which had served me for deciphering the manuscript had enabled me to learn the word — but on a caprice it struck me to tell her that a genie had revealed it to me. This false disclosure fettered Madame d'Urfé to me. That day I became the master of her soul, and I abused my power. Every time I think of it, I am distressed and ashamed, and I do penance now in the obligation under which I place myself of telling the truth in writing my memoirs.

— Casanova, 1757, quoted in D. Kahn's *The Codebreakers*

The situation persisted for another 220 years after this encounter between Casanova and Madame d'Urfé: knowledge of how to encipher and knowledge of how to decipher were regarded as essentially equivalent in any cryptosystem. However, in 1976 W. Diffie and M. Hellman discovered an entirely different type of cryptosystem and invented "public key cryptography."

By definition, a public key cryptosystem has the property that someone who knows only how to encipher cannot use the enciphering key to find

the deciphering key without a prohibitively lengthy computation. In other words the enciphering function $f: \mathcal{P} \longrightarrow \mathcal{C}$ is easy to compute once the enciphering key K_E is known, but it is very hard in practice to compute the inverse function $f^{-1}: \mathcal{C} \longrightarrow \mathcal{P}$. That is, from the standpoint of realistic computability, the function f is not invertible (without some additional information — the deciphering key K_D). Such a function f is called a *trapdoor function*. That is, a trapdoor function f is a function which is easy to compute but whose inverse f^{-1} is hard to compute without having some additional auxiliary information beyond what is necessary to compute f. The inverse f^{-1} is easy to compute, however, for someone who has this information K_D (the "deciphering key").

There is a closely related concept of a *one-way* function. This is a function f which is easy to compute but for which f^{-1} is hard to compute and cannot be made easy to compute even by acquiring some additional information. While the notion of a trapdoor function apparently appeared for the first time in 1978 along with the invention of the RSA public-key cryptosystem, the notion of a one-way function is somewhat older. What seems to have been the first use of one-way functions for cryptography was described in Wilkes' book about time-sharing systems that was published in 1968. The author describes a new *one-way cipher* used by R. M. Needham in order to make it possible for a computer to verify passwords without storing information that could be used by an intruder to impersonate a legitimate user.

> In Needham's system, when the user first sets his password, or whenever he changes it, it is immediately subjected to the enciphering process, and it is the enciphered form that is stored in the computer. Whenever the password is typed in response to a demand from the supervisor for the user's identity to be established, it is again enciphered and the result compared with the stored version. It would be of no immediate use to a would-be malefactor to obtain a copy of the list of enciphered passwords, since he would have to decipher them before he could use them. For this purpose, he would need access to a computer and even if full details of the enciphering algorithm were available, the deciphering process would take a long time.

In 1974, G. Purdy published the first detailed description of such a one-way function. The original passwords and their enciphered forms are regarded as integers modulo a large prime p, and the "one-way" map $\mathbf{F}_p \longrightarrow \mathbf{F}_p$ is given by a polynomial $f(x)$ which is not hard to evaluate by computer but which takes an unreasonably long time to invert. Purdy used $p = 2^{64} - 59$, $f(x) = x^{2^{24}+17} + a_1 x^{2^{24}+3} + a_2 x^3 + a_3 x^2 + a_4 x + a_5$, where the coefficients a_i were arbitrary 19-digit integers.

The above definitions of a public key cryptosystem and a one-way or trapdoor function are not precise from a rigorous mathematical standpoint. The notion of "realistic computability" plays a basic role. But that is an empirical concept that is affected by advances in computer technology (e.g., parallel processor techniques) and the discovery of new algorithms which speed up the performance of arithmetic tasks (sometimes by a large factor). Thus, it is possible that an enciphering transformation that can safely be regarded as a one-way or trapdoor function in 1994 might lose its one-way or trapdoor status in 2004 or in the year 2994.

It is conceivable that some transformation could be *proved* to be trapdoor. That is, there could be a theorem that provides a nontrivial lower bound for the number of bit operations that would be required ("on the average," i.e., for random values of the key parameters) in order to figure out and implement a deciphering algorithm without the deciphering key. Here one would have to allow the possibility of examining a large number of corresponding plaintext–ciphertext message units (as in our frequency analysis of the simple systems in Chapter III), because, by the definition of a public key system, any user can generate an arbitrary number of plaintext–ciphertext pairs. One would also have to allow the use of "probabilistic" methods which, while not guaranteed to break the code at once, would be likely to work if repeated many times. (Examples of probabilistic algorithms will be given in the next chapter.) Unfortunately, no such theorems have been proved for any of the functions that have been used as enciphering maps. Thus, while there are now many cryptosystems which empirically seem to earn the right to be called "public key," there is no cryptosystem in existence which is *provably* public key.

The reason for the name "public key" is that the information needed to send secret messages — the enciphering key K_E — can be made public information without enabling anyone to read the secret messages. That is, suppose we have some population of users of the cryptosystem, each one of whom wants to be able to receive confidential communications from any of the other users without a third party (either another user or an outsider) being able to decipher the message. Some central office can collect the enciphering key $K_{E,A}$ from each user A and publish all of the keys in a "telephone book" having the form

AAA Banking Company	(9974398087453939, 2975290017591012)
Aardvark, Aaron	(8870004228331, 7234752637937)
⋮	⋮

Someone wanting to send a message merely has to look up the enciphering key in this "telephone book" and then use the general enciphering algorithm with the key parameters corresponding to the intended recipient. Only the intended recipient has the matching deciphering key needed to read the message.

In earlier ages this type of system would not have seemed to have any particularly striking advantages. Traditionally, cryptography was used mainly for military and diplomatic purposes. Usually there was a small, well-defined group of users who could all share a system of keys, and new keys could be distributed periodically (using couriers) so as to keep the enemy guessing.

However, in recent years the actual and potential applications of cryptography have expanded to include many other areas where communication systems play a vital role — collecting and keeping records of confidential data, electronic financial transactions, and so on. Often one has a large network of users, any two of whom should be able to keep their communications secret from all other users as well as intruders from outside the network. Two parties may share a secret communication on one occasion, and then a little later one of them may want to send a confidential message to a third party. That is, the "alliances" — who is sharing a secret with whom — may be continually shifting. It might be impractical always to be exchanging keys with all possible confidential correspondents.

Notice that with a public key system it is possible for two parties to initiate secret communications without ever having had any prior contact, without having established any prior trust for one another, without exchanging any preliminary information. All of the information necessary to send an enciphered message is publicly available.

Classical vesus public key. By a *classical* cryptosystem (also called a *private key* cryptosystem or a *symmetrical* cryptosystem), we mean a cryptosystem in which, once the enciphering information is known, the deciphering transformation can be implemented in approximately the same order of magnitude of time as the enciphering transformation. All of the cryptosystems in Chapter III are classical. Occasionally, it takes a little longer for the deciphering — because one needs to apply the Euclidean algorithm to find an inverse modulo N or one must invert a matrix (and this can take a fairly long time if we work with $k \times k$ -matrices for k larger than 2) — nevertheless, the additional time required is not prohibitive. (Moreover, usually the additional time is required only once — to find K_D — after which it takes no longer to decipher than to encipher.) For example, we might need only $O(log^2 B)$ to encipher a message unit, and $O(log^3 B)$ bit operations to decipher one by finding K_D from K_E, where B is a bound on the size of the key parameters. Notice the role of big-O estimates here.

If, on the other hand, the enciphering time were polynomial in $log\ B$ and the deciphering time (based on knowledge of K_E but not K_D) were, say, polynomial in B but not in $\log B$, then we would have a *public key* rather than a classical cryptosystem.

Authentication. Often, one of the most important parts of a message is the *signature*. A person's signature — hopefully, written with an idiosyncratic flourish of the pen which is hard to duplicate — lets the recipient know that the message really is from the person whose name is typed below. If the message is particularly important, it might be necessary to use additional methods to *authenticate* the communication. And in electronic communication, where one does not have a physical signature, one has to rely entirely on other methods. For example, when an officer of a corporation wants to withdraw money from the corporate account by telephone, he/she is often asked to give some personal information (e.g., mother's maiden name) which the corporate officer knows and the bank knows (from data submitted when the account was opened) but which an imposter would not be likely to know.

In public key cryptography there is an especially easy way to identify oneself in such a way that no one could be simply pretending to be you. Let A (Alice) and B (Bob) be two users of the system. Let f_A be the enciphering transformation with which any user of the system sends a message to Alice, and let f_B be the same for Bob. For simplicity, we shall assume that the set $\mathcal{P}$ of all possible plaintext message units and the set $\mathcal{C}$ of all possible ciphertext message units are equal, and are the same for all users. Let P be Alice's "signature" (perhaps including an identification number, a statement of the time the message was sent, etc.). It would not be enough for Alice to send Bob the encoded message $f_B(P)$, since *everyone* knows how to do that, so there would be no way of knowing that the signature was not

forged. Rather, at the beginning (or end) of the message Alice transmits $f_B f_A^{-1}(P)$. Then, when Bob deciphers the whole message, including this part, by applying f_B^{-1}, he finds that everything has become plaintext except for a small section of jibberish, which is $f_A^{-1}(P)$. Since Bob knows that the message is claimed to be from Alice, he applies f_A (which he knows, since Alice's enciphering key is public), and obtains P. Since no one other than Alice could have applied the function f_A^{-1} which is inverted by f_A, he knows that the message was from Alice.

Hash functions. A common way to sign a document is with the help of a *hash function*. Roughly speaking, a hash function is an easily computable map $f : x \mapsto h$ from a very long input x to a much shorter output h (for example, from strings of about 10^6 bits to strings of 150 or 200 bits) that has the following property: *it is not computationally feasible to find two different inputs x and x' such that $f(x') = f(x)$*. If part of Alice's "signature" consists of the hash value $h = f(x)$, where x is the entire text of her message, then Bob can verify not only that the message was really sent by Alice, but also that it wasn't tampered with during transmission. Namely, Bob applies the hash function f to his deciphered plaintext from Alice, and checks that the result agrees with the value h in Alice's signature. By assumption, no tamperer would have been able to change x without changing the value $h = f(x)$.

Key exchange. In practice, the public key cryptosystems for sending messages tend to be slower to implement than the classical systems that are in current use. The number of plaintext message units per second that can be transmitted is less. However, even if a network of users feels attached

to the traditional type of cryptosystem, they may want to use a public key cryptosystem in an auxiliary capacity to send one another their keys $K = (K_E, K_D)$ for the classical system. Thus, the ground rules for the classical cryptosystem can be agreed upon, and keys can be periodically exchanged, using the slower public key cryptography; while the large volume of messages would then be sent by the faster, older methods.

Probabilistic Encryption. Most of the number theory based cryptosystems for message transmission are *deterministic*, in the sense that a given plaintext will always be encrypted into the same ciphertext any time it is sent. However, deterministic encryption has two disadvantages: (1) if an eavesdropper knows that the plaintext message belongs to a small set (for example, the message is either "yes" or "no"), then she can simply encrypt all possibilities in order to determine which is the supposedly secret message; and (2) it seems to be very difficult to *prove* anything about the security of a system if the encryption is deterministic. For these reasons, *probabilistic encryption* was introduced. We will not discuss this further or give examples in this book. For more information, see the fundamental papers on the subject by Goldwasser and Micali (*Proc. 14th ACM Symp. Theory of Computing*, 1982, 365–377, and *J. Comput. System Sci.* **28** (1984), 270–299).

2 RSA

In looking for a trapdoor function f to use for a public key cryptosystem, one wants to use an idea which is fairly simple conceptually and lends itself to easy implementation. On the other hand, one wants to have very strong empirical evidence — based on a long history of attempts to find algorithms for f^{-1} — that decryption cannot feasibly be accomplished without knowledge of the secret deciphering key. For this reason it is natural to look at an ancient problem of number theory: the problem of finding the complete factorization of a large composite integer whose prime factors are not known in advance. The success of the so-called "RSA" cryptosystem (from the last names of the inventors Rivest, Shamir, and Adleman), which is one of the oldest (16 years old) and most popular public key cryptosystems, is based on the tremendous difficulty of factoring.

We now describe how RSA works. Each user first chooses two extremely large prime numbers p and q (say, of about 100 decimal digits each), and sets $n = pq$. Knowing the factorization of n, it is easy to compute $\varphi(n) = (p-1)(q-1) = n+1-p-q$. Next, the user randomly chooses an integer e between 1 and $\varphi(n)$ which is prime to $\varphi(n)$.

Remark. Whenever we say "random" we mean that the number was chosen with the help of a random-number generator (or "pseudo-random" number generator), i.e., a computer program that generates a sequence of digits in a way that no one could duplicate or predict, and which is likely to have all of the statistical properties of a truly random sequence. A lot has been written concerning efficient and secure ways to generate random numbers, but we shall not concern ourselves with this question here. In the RSA cryptosystem we need a random number generator not only to choose e, but also to choose the large primes p and q (so that no one could guess our choices by looking at tables of special types of primes, for example, Mersenne primes or factors of $b^k \pm 1$ for small b and relatively small k). What does a "randomly generated" prime number mean? Well, first generate a large random integer m. If m is even, replace m by $m+1$. Then apply suitable *primality tests* to see if the odd number m is prime (primality tests will be examined systematically in the next chapter). If m is not prime, try $m+2$, then $m+4$, and so on, until you reach the first prime number $\geq m$, which is what you take as your "random" prime. According to the Prime Number Theorem (for the statement see Exercise 13 of § I.1), the frequency of primes among the numbers near m is about $1/log(m)$, so you can expect to test $O(log\, m)$ numbers for primality before reaching the first prime $\geq m$.

Similarly, the "random" number e prime to $\varphi(n)$ can be chosen by first generating a random (odd) integer with an appropriate number of bits, and then successively incrementing it until one finds an e with $g.c.d.(e, \varphi(n)) = 1$. (Alternately, one can perform primality tests until one finds a prime e, say between $max(p, q)$ and $\varphi(n)$; such a prime must necessarily satisfy $g.c.d.(e, \varphi(n)) = 1$.)

Thus, each user A chooses two primes p_A and q_A and a random number e_A which has no common factor with $(p_A - 1)(q_A - 1)$. Next, A computes $n_A = p_A q_A$, $\varphi(n_A) = n_A + 1 - p_A - q_A$, and also the multiplicative inverse of e_A modulo $\varphi(n_A)$: $d_A \underset{\text{def}}{=} e_A^{-1} \ mod\ \varphi(n_A)$. She makes public the enciphering key $K_{E,A} = (n_A, e_A)$ and conceals the deciphering key $K_{D,A} = (n_A, d_A)$. The enciphering transformation is the map from $\mathbf{Z}/n_A\mathbf{Z}$ to itself given by $f(P) \equiv P^{e_A} \ mod\ n_A$. The deciphering transformation is the map from $\mathbf{Z}/n_A\mathbf{Z}$ to itself given by $f^{-1}(C) \equiv C^{d_A} \ mod\ n_A$. It is not hard to see that these two maps are inverse to one another, because of our choice of d_A. Namely, performing f followed by f^{-1} or f^{-1} followed by f means raising to the $d_A e_A$-th power. But, because $d_A e_A$ leaves a remainder of 1 when divided by $\varphi(n_A)$, this is the same as raising to the 1-st power (see the corollary of Proposition I.3.5, which gives this in the case when P has no common factor with n_A; if $g.c.d.(P, n_A) > 1$, see Exercise 6 below).

From the description in the last paragraph, it seems that we are working with sets $\mathcal{P} = \mathcal{C}$ of plaintext and ciphertext message units that vary from one user to another. In practice, we would probably want to choose $\mathcal{P}$ and $\mathcal{C}$ uniformly throughout the system. For example, suppose we are working in an N-letter alphabet. Then let $k < \ell$ be suitably chosen positive integers, such that, for example, N^k and N^ℓ have approximately 200 decimal digits. We take as our plaintext message units all blocks of k letters, which we regard as k-digit base-N integers, i.e., we assign them numerical equivalents between 0 and N^k. We similarly take ciphertext message units to be blocks of ℓ letters in our N-letter alphabet. Then each user must choose his/her large primes p_A and q_A so that $n_A = p_A q_A$ satisfies $N^k < n_A < N^\ell$. Then any plaintext message unit, i.e., integer less than N^k, corresponds to

an element in $\mathbf{Z}/n_A\mathbf{Z}$ (for any user's n_A); and, since $n_A < N^\ell$, the image $f(P) \in \mathbf{Z}/n_A\mathbf{Z}$ can be uniquely written as an ℓ-letter block. (Not all ℓ-letter blocks can arise — only those corresponding to integers less than n_A for the particular user's n_A.)

Example 1. For the benefit of a reader who doesn't have a computer handy (or does not have good multiple precision software), we shall sacrifice realism and choose most of our examples so as to involve relatively small integers. Choose $N = 26$, $k = 3$, $\ell = 4$. That is, the plaintext consists of trigraphs and the ciphertext consists of four-graphs in the usual 26-letter alphabet. To send the message "YES" to a user A with enciphering key $(n_A, e_A) = (46927, 39423)$, we first find the numerical equivalent of "YES," namely: $24 \cdot 26^2 + 4 \cdot 26 + 18 = 16346$, and then compute 16346^{39423} *mod* 46927, which is $21166 = 1 \cdot 26^3 + 5 \cdot 26^2 + 8 \cdot 26 + 2 =$"BFIC."

The recipient A knows the deciphering key $(n_A, d_A) = (46927, 26767)$, and so computes $21166^{26767} \ mod\ 46927 = 16346$ ="YES." How did user A generate her keys? First, she multiplied the primes $p_A = 281$ and $q_A = 167$ to get n_A; then she chose e_A at random (but subject to the condition that $g.c.d.(e_A, 280) = g.c.d.(e_A, 166) = 1$). Then she found $d_A = e_A^{-1} \ mod\ 280 \cdot 166$. The numbers p_A, q_A, d_A remain secret.

In Example 1, how cumbersome are the computations? The most time-consuming step is modular exponentiation, e.g., $16346^{39423} \ mod\ 46927$. But this can be done by the repeated squaring method (see §I.3) in $O(k^3)$ bit operations, where k is the number of bits in our integers. Actually, if we were working with much larger integers, potentially the most time-consuming step would be for each user A to find two very large primes p_A and q_A. In order to quickly choose suitable very large primes, one must use an efficient primality test. Such tests will be described in the next chapter.

Remarks. **1.** In choosing p and q, user A should take care to see that certain conditions hold. The most important are: that the two primes not be too close together (for example, one should be a few decimal digits longer than the other); and that $p-1$ and $q-1$ have a fairly small g.c.d. and both have at least one large prime factor. Some of the reasons for these conditions are indicated in the exercises below. Of course, if someone discovers a factorization method that works quickly under certain other conditions on p and q, then future users of RSA would have to take care to avoid those conditions as well.

2. In §I.3 we saw that, when n is a product of two primes p and q, knowledge of $\varphi(n)$ is equivalent to knowledge of the factorization. Let's suppose now that we manage to break an RSA system by determining a positive integer d such that $a^{ed} \equiv a \ mod\ n$ for all a prime to n. This is equivalent to $ed-1$ being a multiple of the least common multiple of $p-1$ and $q-1$. Knowing this integer $m = ed-1$ is weaker than actually knowing $\varphi(n)$. But we now give a procedure that with a high probability is nevertheless able to use the integer m to factor n.

So suppose we know n — which is a product of two unknown primes — and also an integer m such that $a^m \equiv 1 \ mod\ n$ for all a prime to n. Notice that any such m must be even (as we see by taking $a = -1$). We first check whether $m/2$ has the same property, in which case we can replace m by $m/2$. If $a^{m/2}$ is *not* $\equiv 1 \ mod\ n$ for all a prime to n, then we

must have $a^{m/2} \not\equiv 1 \ mod\ n$ for at least 50% of the a's in $(\mathbf{Z}/n\mathbf{Z})^*$ (this statement is proved in exactly the same way as part (a) of Exercise 21 in §II.2). Thus, if we test several dozen randomly chosen a's and find that in all cases $a^{m/2} \equiv 1 \ mod\ n$, then with very high probability we have this congruence for all a prime to n, and so may replace m by $m/2$. We keep on doing this until we no longer have the congruence when we take half of the exponent. There are now two possibilities:

(i) $m/2$ is a multiple of one of the two numbers $p-1$, $q-1$ (say, $p-1$) but not both. In this case $a^{m/2}$ is always $\equiv 1 \ mod\ p$ but exactly 50% of the time is congruent to -1 rather than $+1$ modulo q.

(ii) $m/2$ is not a multiple of either $p-1$ or $q-1$. In this case $a^{m/2}$ is $\equiv 1$ modulo both p and q (and hence modulo n) exactly 25% of the time, it is $\equiv -1$ modulo both p and q exactly 25% of the time, and for the remaining 50% of the values of a it is $\equiv 1$ modulo one of the primes and $\equiv -1$ modulo the other prime.

Thus, by trying a's at random with high probability we will soon find an a for which $a^{m/2}-1$ is divisible by one of the two primes (say, p) but not the other. (Each randomly selected a has a 50% chance of satisfying this statement.) Once we find such an a we can immediately factor n, because $g.c.d.(n,\ a^{m/2}-1) = p$.

The above procedure is an example of a *probabilistic algorithm*. We shall encounter other probabilistic algorithms in the next chapter.

3. How do we send a signature in RSA? When discussing authentication in the last section, we assumed for simplicity that $\mathcal{P} = \mathcal{C}$. We have a slightly more complicated set-up in RSA. Here is one way to avoid the problem of different n_A's and different block sizes (k, the number of letters in a plaintext message unit, being less than ℓ, the number of letters in a ciphertext message unit). Suppose that, as in the last section, Alice is sending her signature (some plaintext P) to Bob. She knows Bob's enciphering key $K_{E,B} = (n_B, e_B)$ and her own deciphering key $K_{D,A} = (n_A, d_A)$. What she does is send $f_B f_A^{-1}(P)$ if $n_A < n_B$, or else $f_A^{-1} f_B(P)$ if $n_A > n_B$. That is, in the former case she takes the least positive residue of P^{d_A} modulo n_A; then, regarding that number modulo n_B, she computes $(P^{d_A} \ mod\ n_A)^{e_B} \ mod\ n_B$, which she sends as a ciphertext message unit. In the case $n_A > n_B$, she first computes $P^{e_B} \ mod\ n_B$ and then, working modulo n_A, she raises this to the d_A-th power. Clearly, Bob can verify the authenticity of the message

in the first case by raising to the d_B-th power modulo n_B and then to the e_A-th power modulo n_A; in the second case he does these two operations in the reverse order.

THEOREM 11-3 (Fermat). *The Diophantine equation* $x^4 + y^4 = z^2$ *has no solution in positive integers* x, y, z.

Proof: With the idea of deriving a contradiction, let us assume that there exists a positive solution x_0, y_0, z_0 of $x^4 + y^4 = z^2$. Nothing is lost in supposing also that $\gcd(x_0, y_0) = 1$; otherwise, put $\gcd(x_0, y_0) = d$, $x_0 = dx_1, y_0 = dy_1, z_0 = d^2 z_1$ to get ${x_1}^4 + {y_1}^4 = {z_1}^2$ with $\gcd(x_1, y_1) = 1$.

Expressing the supposed equation ${x_0}^4 + {y_0}^4 = {z_0}^2$ in the form

$$({x_0}^2)^2 + ({y_0}^2)^2 = {z_0}^2$$

we see that ${x_0}^2, {y_0}^2, z_0$ meet all the requirements of a primitive Pythagorean triple, and so Theorem 11-1 can be brought into play.

In such triples, one of the integers x_0^2 or y_0^2 is necessarily even, while the other is odd. Taking x_0^2 (and hence x_0) to be even, there exist relatively prime integers $s > t > 0$ satisfying

$$x_0^2 = 2st,$$
$$y_0^2 = s^2 - t^2,$$
$$z_0 = s^2 + t^2,$$

where exactly one of s and t is even. If it happened that s were even, then we would have

$$1 \equiv y_0^2 = s^2 - t^2 \equiv 0 - 1 \equiv 3 \pmod 4,$$

an impossibility. Therefore, s must be the odd integer and, in consequence, t is the even one. Let us put $t = 2r$. Then the equation $x_0^2 = 2st$ becomes $x_0^2 = 4sr$, which says that

$$(x_0/2)^2 = sr.$$

But Lemma 2 asserts that the product of two relatively prime integers [gcd $(s, t) = 1$ implies that gcd $(s, r) = 1$] is a square only if each of the integers is itself a square; hence, $s = z_1^2$, $r = w_1^2$ for positive integers z_1, w_1.

We wish to apply Theorem 11-1 again, this time to the equation

$$t^2 + y_0^2 = s^2.$$

Since gcd $(s, t) = 1$, it follows that gcd $(t, y_0, s) = 1$, making t, y_0, s a primitive Pythagorean triple. With t even, we obtain

$$t = 2uv,$$
$$y_0 = u^2 - v^2,$$
$$s = u^2 + v^2,$$

for relatively prime integers $u > v > 0$. Now the relation

$$uv = t/2 = r = w_1^2$$

signifies that u and v are both squares (Lemma 2 serves its purpose once more); say, $u = x_1^2$ and $v = y_1^2$. When these values are substituted into the equation for s the result is

$$z_1^2 = s = u^2 + v^2 = x_1^4 + y_1^4.$$

A crucial point is that, z_1 and t being positive, we also have the inequality

$$0 < z_1 \leq z_1^2 = s \leq s^2 < s^2 + t^2 = z_0.$$

What has happened is this: starting with one solution x_0, y_0, z_0 of $x^4 + y^4 = z^2$, we have constructed another solution x_1, y_1, z_1 such that $0 < z_1 < z_0$. Repeating the whole argument, our second solution would lead to a third solution x_2, y_2, z_2 with $0 < z_2 < z_1$, which in its turn gives rise to a fourth. This process can be carried out indefinitely to produce an infinite decreasing sequence of positive integers

$$z_0 > z_1 > z_2 > \cdots.$$

Since there is only a finite supply of positive integers less than z_0, a contradiction occurs. We are forced to conclude that $x^4 + y^4 = z^2$ is not solvable in the positive integers.

COROLLARY. *The equation* $x^4 + y^4 = z^4$ *has no solution in the positive integers.*

Proof: If x_0, y_0, z_0 were a positive solution of $x^4 + y^4 = z^4$, then x_0, y_0, z_0^2 would satisfy the equation $x^4 + y^4 = z^2$, in conflict with Theorem 11-3.

If $n > 2$, then n is either a power of 2 or divisible by an odd prime p. In the first case, $n = 4k$ for some $k \geq 1$ and the Fermat equation $x^n + y^n = z^n$ can be written as

$$(x^k)^4 + (y^k)^4 = (z^k)^4.$$

We have just seen that this equation is impossible in the positive integers. When $n = pk$, the Fermat equation is the same as

$$(x^k)^p + (y^k)^p = (z^k)^p.$$

If it could be shown that the equation $u^p + v^p = w^p$ has no solution, then, in particular, there would be no solution of the form $u = x^k$, $v = y^k$, $w = z^k$ and hence $x^n + y^n = z^n$ would not be solvable. Fermat's Conjecture therefore reduces to this: for no odd prime p does the equation

$$x^p + y^p = z^p$$

admit a solution in the positive integers.

Although the problem has challenged the foremost mathematicians of the last 300 years, their efforts have only produced partial results and proofs of individual cases. Euler gave the first proof of the Fermat Conjecture for the prime $p = 3$ in the year 1770; the reasoning was incomplete at one stage, but Legendre later supplied the missing steps. Using the method of infinite descent, Dirichlet and Legendre independently settled the case $p = 5$ around 1825. Not long thereafter, in 1839, Lamé proved the conjecture for seventh powers. With the increasing complexity of the arguments came the realization that a successful resolution of the general case called for different techniques. The best hope seemed to lie in extending the meaning of "integer" to include a wider class of numbers and, by attacking the problem within this enlarged system, obtaining more information than was possible by using ordinary integers only.

The German mathematician Kummer made the major breakthrough. In 1843, he submitted to Dirichlet a purported proof of the Fermat Conjecture based upon an extension of the integers to include the so-called "algebraic numbers" (that is, complex numbers satisfying polynomials with rational coefficients). Having spent considerable time on the problem himself, Dirichlet was immediately able to detect the flaw in the reasoning: Kummer had taken for granted that algebraic numbers admit a unique factorization similar to that of the ordinary integers, and this is not always true.

But Kummer was undeterred by this perplexing situation and returned to his investigations with redoubled effort. In order to restore unique factorization to the algebraic numbers, he was led to invent the concept of *ideal numbers*. By adjoining these new entities to the algebraic numbers, Kummer successfully proved the Fermat Conjecture for a large class of primes which he termed "regular primes" (that this repre-

sented an enormous achievement is reflected in the fact that the only irregular primes less than 100 are 37, 59, and 67.). Unfortunately, it is still not known whether there are an infinite number of regular primes, while, in the other direction, Jensen (1915) established that there exist infinitely many irregular ones. Almost all the subsequent progress on the problem has been within the framework suggested by Kummer.

To round out our historical digression, we might mention that in 1908 a prize of 100,000 marks was bequeathed to the Academy of Science at Göttingen to be paid for the first complete proof of Fermat's Conjecture. The immediate result was a deluge of incorrect demonstrations by amateur mathematicians. Since only printed solutions were eligible, Fermat's Conjecture is reputed to be the mathematical problem for which the greatest number of false proofs have been published; indeed, between 1908 and 1912 over 1000 alleged proofs appeared, mostly printed as private pamphlets. Suffice it to say, interest declined as the German inflation of the 1920's wiped out the monetary value of the prize.

From $x^4 + y^4 = z^2$, we move on to a closely related Diophantine equation, namely, $x^4 - y^4 = z^2$. The proof of its insolubility parallels that of Theorem 11-3, but we give a slight variation in the method of infinite descent.

THEOREM 11-4 (Fermat). *The Diophantine equation* $x^4 - y^4 = z^2$ *has no solution in positive integers* x, y, z.

Proof: The proof proceeds by contradiction. Let us assume that the equation admits a solution in the positive integers and among these solutions x_0, y_0, z_0 is one with a least value of x; in particular, this supposition forces x_0 to be odd (Why?). Were $\gcd(x_0, y_0) = d > 1$,

then putting $x_0 = dx_1, y_0 = dy_1$, we would have $d^4(x_1^4 - y_1^4) = z_0^2$, whence $d^2 \mid z_0$ or $z_0 = d^2 z_1$ for some $z_1 > 0$. It follows that x_1, y_1, z_1 provides a solution to the equation under consideration with $0 < x_1 < x_0$, an impossible situation. Thus, we are free to assume a solution x_0, y_0, z_0 in which $\gcd(x_0, y_0) = 1$. The ensuing argument falls into two stages, depending on whether y_0 is odd or even.

First, consider the case of an odd integer y_0. If the equation $x_0^4 - y_0^4 = z_0^2$ is written in the form $z_0^2 + (y_0^2)^2 = (x_0^2)^2$, one sees that z_0, y_0^2, x_0^2 constitute a primitive Pythagorean triple. Theorem 11-1 asserts the existence of relatively prime integers $s > t > 0$ for which

$$z_0 = 2st,$$
$$y_0^2 = s^2 - t^2,$$
$$x_0^2 = s^2 + t^2.$$

It thus appears that

$$s^4 - t^4 = (s^2 + t^2)(s^2 - t^2) = x_0^2 y_0^2 = (x_0 y_0)^2,$$

making s, t, $x_0 y_0$ a (positive) solution to the equation $x^4 - y^4 = z^2$. Since

$$0 < s < \sqrt{s^2 + t^2} = x_0,$$

we arrive at a contradiction to the minimal nature of x_0.

For the second part of the proof, assume that y_0 is an even integer. Using the formulas for primitive Pythagorean triples, we now write

$$y_0^2 = 2st,$$
$$z_0 = s^2 - t^2,$$
$$x_0^2 = s^2 + t^2,$$

It thus appears that

$$s^4 - t^4 = (s^2 + t^2)(s^2 - t^2) = x_0^2 y_0^2 = (x_0 y_0)^2,$$

making s, t, $x_0 y_0$ a (positive) solution to the equation $x^4 - y^4 = z^2$.

Since

$$0 < s < \sqrt{s^2 + t^2} = x_0,$$

we arrive at a contradiction to the minimal nature of x_0.

For the second part of the proof, assume that y_0 is an even integer. Using the formulas for primitive Pythagorean triples, we now write

$$y_0^2 = 2st,$$
$$z_0 = s^2 - t^2,$$
$$x_0^2 = s^2 + t^2,$$

where s may be taken to be even and t to be odd. Then, in the relation $y_0^2 = 2st$, we have gcd $(2s, t) = 1$. The by-now-customary Lemma 2 tells us that $2s$ and t are each squares of positive integers; say, $2s = w^2$, $t = v^2$. Since w must of necessity be an even integer, set $w = 2u$ to get $s = 2u^2$. Therefore,

$$x_0^2 = s^2 + t^2 = 4u^4 + v^4$$

and so $2u^2$, v^2, x_0 forms a primitive Pythagorean triple. Falling back on Theorem 11-1 again, there exist integers $a > b > 0$ for which

$$2u^2 = 2ab,$$
$$v^2 = a^2 - b^2,$$
$$x_0 = a^2 + b^2,$$

where gcd $(a, b) = 1$. The equality $u^2 = ab$ ensures that a and b are perfect squares, so that $a = c^2$ and $b = d^2$. Knowing this, the rest of the proof is easy; for, upon substituting,

$$v^2 = a^2 - b^2 = c^4 - d^4.$$

The result is a new solution c, d, v of the given equation $x^4 - y^4 = z^2$ and what's more, a solution in which

$$0 < c = \sqrt{a} < a^2 + b^2 = x_0,$$

contrary to our assumption regarding x_0.

The only resolution of these contradictions is that the equation $x^4 - y^4 = z^2$ cannot be satisfied in the positive integers.

THEOREM 11-5. *The area of a Pythagorean triangle can never be equal to a perfect (integral) square.*

Proof: Consider a Pythagorean triangle whose hypotenuse has length z and other two sides have lengths x and y, so that $x^2 + y^2 = z^2$. The area of the triangle in question is $\frac{1}{2}xy$ and if this were a square, say u^2, it would follow that $2xy = 4u^2$. By adding and subtracting the last-written equation from $x^2 + y^2 = z^2$, we are led to

$$(x + y)^2 = z^2 + 4u^2 \quad \text{and} \quad (x - y)^2 = z^2 - 4u^2.$$

When these last two equations are multiplied together, the outcome is that two fourth powers have as their difference a square:

$$(x^2 - y^2)^2 = z^4 - 16u^4 = z^4 - (2u)^4.$$

Since this amounts to an infringement of Theorem 11-4, there can be no Pythagorean triangle whose area is a square.

There are a number of simple problems pertaining to Pythagorean triangles that still await solution. The Corollary to Theorem 11-3 may be expressed by saying that there exists no Pythagorean triangle all the sides of which are squares. However, it is not difficult to produce Pythagorean triangles whose sides, if increased by 1, are squares; for instance, the triangles associated with the triples $13^2 - 1$, $10^2 - 1$, $14^2 - 1$, and $287^2 - 1$, $265^2 - 1$, $329^2 - 1$. An obvious—and as yet unanswered—question is whether there are an infinite number of such triangles. One can find Pythagorean triangles each side of which is a triangular number. [By a triangular number, we mean an integer of the form $t_n = n(n+1)/2$.] An example of such is the triangle corresponding to $t_{132}, t_{143}, t_{164}$. It is not known if there exist infinitely many Pythagorean triangles of this type.

As a closing comment, we should observe that all the effort expended on attempting to prove Fermat's Conjecture has been far from wasted. The new mathematics that was developed as a by-product laid the foundations for algebraic number theory, as well as the ideal theory of modern abstract algebra. It seems fair to say that the value of these far exceeds that of the conjecture itself.

PROBLEMS

1. Show that the equation $x^2 + y^2 = z^3$ has infinitely many solutions for x, y, z positive integers. [*Hint:* For any $n > 3$, let $x = n(n^2 - 3)$ and $y = 3n^2 - 1$.]
2. Prove the theorem: The only solutions in nonnegative integers of the equation $x^2 + 2y^2 = z^2$, with gcd $(x, y, z) = 1$, are given by

$$x = \pm(2s^2 - t^2),\ y = 2st,\ z = 2s^2 + t^2$$

where s, t are arbitrary nonnegative integers. [*Hint:* If u, v, w are such that $y = 2w$, $z + x = 2u$, $z - x = 2v$, then the equation becomes $2w^2 = uv$.]