CSC203: DISCRETE STRUCTURES (2 Units C: LH 45)

Propositional Logic. Predicate Logic. Sets. Functions. Sequences and Summation. Proof Techniques. Mathematical induction. Inclusion-exclusion and Pigeonhole principles. Permutations and Combinations (with and without repetitions). The Binomial Theorem. Discrete Probability. Recurrence Relations.

Learning Outcomes

The students should be able to:

1. convert logical statements from informal language to propositional and predicate logic expressions;

2. describe the strengths and limitations of propositional and predicate logic;

3. outline the basic structure of each proof technique (direct proof, proof by contradiction, and induction) described in this unit;

4. apply each of the proof techniques (direct proof, proof by contradiction, and induction) correctly in the construction of a sound argument;

5. apply the pigeonhole principle in the context of a formal proof.;

6. compute permutations and combinations of a set, and interpret the meaning in the context of the particular application;

7. map real-world applications to appropriate counting formalisms, such as determining the number of ways to arrange people around a table, subject to constraints on the seating arrangement, or the number of ways to determine certain hands in cards (e.g., a full house); and

8. solve a variety of basic recurrence relations.

Text Book(S):

1. Pace, G. J. (2012). *Mathematics of discrete structures for computer science* (pp. I-XVI). New York: Springer.

Course Writer/Developer: Dr. F.E. AYO, Department of Computer Sciences, Olabisi Onabanjo University, Ago-Iwoye.

E-mail: ayo.femi@oouagoiwoye.edu.ng

1.1 BASIC SET THEORY

A set is a general name for any collection of things or numbers. There must be a way of deciding whether any particular object is a member of the set or not. This may be done by referring to a list of the members of the set or a statement describing them.

For example, A = $\{-3, 3\}=\{x : x^2 = 9\}$.

Note: {x: ...} is read as 'the set of objects x such that . . .'.

1.2 SETS AND ELEMENTS, SUBSETS

1.2.1 Definitions

A set may be viewed as any well-defined collection of objects, called the elements or members of the set. One usually uses capital letters, A, B, X, Y, . . . , to denote sets, and lowercase letters, a, b, x, y, . . ., to denote elements of sets. Synonyms for "set" are "class," "collection," and "family."

Membership in a set is denoted as follows:

- a ∈ S denotes that a belongs to a set S
- a, b ∈ S denotes that a and b belong to a set S

Here ∈ is the symbol meaning "is an element of." We use ∉ to mean "is not an element of."

1.2.2 Specifying Sets

There are essentially two ways to specify a particular set. One way, if possible, is to list its members separated by commas and contained in braces { }. A second way is to state those properties which characterized the elements in the set. Examples illustrating these two ways are:

$$A = \{1, 3, 5, 7, 9\} \text{ and } B = \{x \mid x \text{ is an even integer}, x > 0\}$$

That is, A consists of the numbers 1, 3, 5, 7, 9. The second set, which reads:

B is the set of x such that x is an even integer and x is greater than 0,

denotes the set B whose elements are the positive integers. Note that a letter, usually x, is used to denote a typical member of the set; and the vertical line | is read as "such that" and the comma as "and."

EXAMPLE 1.1

(a) The set A above can also be written as A = {x | x is an odd positive integer, x < 10}.

(b) We cannot list all the elements of the above set B although frequently we specify the set by

$$B = \{2, 4, 6, ...\}$$

where we assume that everyone knows what we mean. Observe that 8 ∈ B, but 3 ∉ B.

(c) Let $E = \{x \mid x^2 - 3x + 2 = 0\}$, F = {2, 1} and G = {1, 2, 2, 1}. Then E = F = G.

We emphasize that a set does not depend on the way in which its elements are displayed. A set remains the same if its elements are repeated or rearranged.

Even if we can list the elements of a set, it may not be practical to do so. That is, we describe a set by listing its elements only if the set contains a few elements; otherwise, we describe a set by the property which characterizes its elements.

1.2.3 Subsets

Suppose every element in a set A is also an element of a set B, that is, suppose $a \in A$ implies $a \in B$. Then A is called a subset of B. We also say that A is contained in B or that B contains A. This relationship is written

$A \subseteq B$ or $B \supseteq A$

Two sets are equal if they both have the same elements or, equivalently, if each is contained in the other. That is:

$A = B$ if and only if $A \subseteq B$ and $B \subseteq A$

If A is not a subset of B, that is, if at least one element of A does not belong to B, we write $A \nsubseteq B$.

EXAMPLE 1.2 Consider the sets:

$A = \{1, 3, 4, 7, 8, 9\}$, $B = \{1, 2, 3, 4, 5\}$, $C = \{1, 3\}$.

Then $C \subseteq A$ and $C \subseteq B$ since 1 and 3, the elements of C, are also members of A and B. But $B \nsubseteq A$ since some of the elements of B, e.g., 2 and 5, do not belong to A. Similarly, $A \nsubseteq B$.

Theorem 1.1: Let A, B, C be any sets. Then:

(i) $A \subseteq A$

(ii) If $A \subseteq B$ and $B \subseteq A$, then $A = B$

(iii) If $A \subseteq B$ and $B \subseteq C$, then $A \subseteq C$

1.2.4 Special symbols

Some sets will occur very often in the course, and so we use special symbols for them. Some such symbols are:

N = the set of natural numbers or positive integers: 1, 2, 3,...

Z = the set of all integers: ..., −2, −1, 0, 1, 2,...

Q = the set of rational numbers

R = the set of real numbers

C = the set of complex numbers

Observe that **N** ⊆ **Z** ⊆ **Q** ⊆ **R**⊆ **C**

1.2.5 Universal Set, Empty Set

All sets under investigation in any application of set theory are assumed to belong to some fixed large set called the universal set which we denote by

U

unless otherwise stated or implied.

Given a universal set U and a property P, there may not be any elements of U which have property P. For example, the following set has no elements:

$S = \{x \mid x \text{ is a positive integer}, x^2 = 3\}$

Such a set with no elements is called the empty set or null set and is denoted by

∅

There is only one empty set. That is, if S and T are both empty, then S = T, since they have exactly the same elements, namely, none.

The empty set ∅ is also regarded as a subset of every other set. Thus we have the following simple result which we state formally.

Theorem 1.2: For any set A, we have ∅ ⊆ A ⊆ U.

1.2.6 Disjoint Sets

Two sets A and B are said to be disjoint if they have no elements in common. For example, suppose

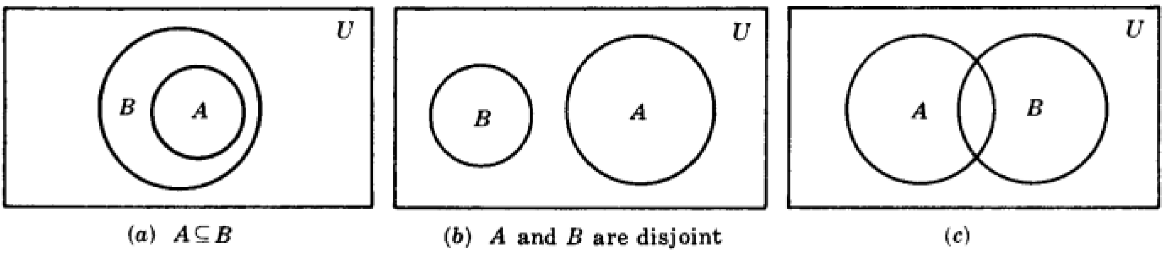
A = {1, 2}, B = {4, 5, 6}, and C = {5, 6, 7, 8}

Then A and B are disjoint, and A and C are disjoint. But B and C are not disjoint since B and C have elements in common, e.g., 5 and 6. We note that if A and B are disjoint, then neither is a subset of the other (unless one is the empty set).

1.3 VENN DIAGRAMS

A Venn diagram is a pictorial representation of sets in which sets are represented by enclosed areas in the plane. The universal set U is represented by the interior of a rectangle, and the other sets are represented by disks lying within the rectangle. If A ⊆ B, then the disk representing A will be entirely within the disk representing B as in Fig. 1-1(a). If A and B are disjoint, then the disk representing A will be separated from the disk representing B as in Fig. 1-1(b).

However, if A and B are two arbitrary sets, it is possible that some objects are in A but not in B, some are in B but not in A, some are in both A and B, and some are in neither A nor B; hence in general we represent A and B as in Fig. 1-1(c).



(a) $A \subseteq B$ (b) A and B are disjoint (c)

Fig. 1-1

1.4 SET OPERATIONS

This unit introduces a number of set operations, including the basic operations of union, intersection, and complement.

1.4.1 Union and Intersection

The union of two sets A and B, denoted by A ∪ B, is the set of all elements which belong to A or to B; that is,

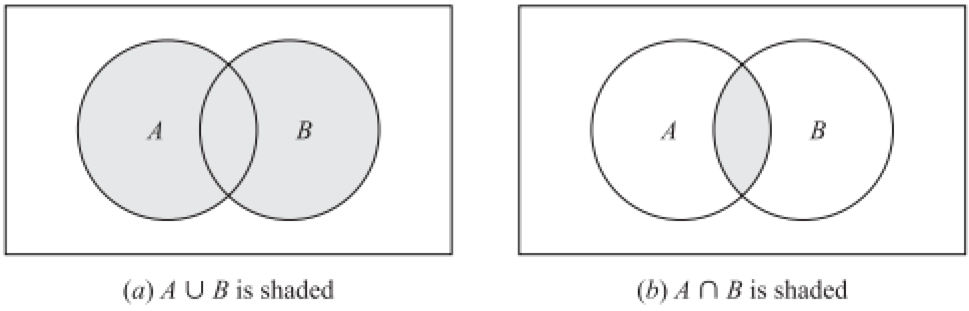
$$A \cup B = \{x \mid x \in A \text{ or } x \in B\}$$

Here “or” is used in the sense of and/or. Figure 1-2(a) is a Venn diagram in which A ∪ B is shaded.

The intersection of two sets A and B, denoted by A ∩ B, is the set of elements which belong to both A and B; that is,

$$A \cap B = \{x \mid x \in A \text{ and } x \in B\}$$

Figure 1-2(b) is a Venn diagram in which A ∩ B is shaded.



(a) $A \cup B$ is shaded (b) $A \cap B$ is shaded

Fig. 1-2

Recall that sets A and B are said to be disjoint or nonintersecting if they have no elements in common or, using the definition of intersection, if A ∩ B = ∅, the empty set. Suppose

$$S = A \cup B \text{ and } A \cap B = \emptyset$$

Then S is called the disjoint union of A and B.

EXAMPLE 1.3

(a) Let A = {1, 2, 3, 4}, B = {3, 4, 5, 6, 7}, C = {2, 3, 8, 9}. Then

$A \cup B = \{1, 2, 3, 4, 5, 6, 7\}$, $A \cup C = \{1, 2, 3, 4, 8, 9\}$, $B \cup C = \{2, 3, 4, 5, 6, 7, 8, 9\}$,

$A \cap B = \{3, 4\}$, $A \cap C = \{2, 3\}$, $B \cap C = \{3\}$.

(b) Let U be the set of students at a university, and let M denote the set of male students and let F denote the set of female students. The U is the disjoint union of M of F; that is,

$U = M \cup F$ and $M \cap F = \emptyset$

This comes from the fact that every student in U is either in M or in F, and clearly no student belongs to both M and F, that is, M and F are disjoint.

Theorem 1.3: For any sets A and B, we have:

(i) $A \cap B \subseteq A \subseteq A \cup B$ and

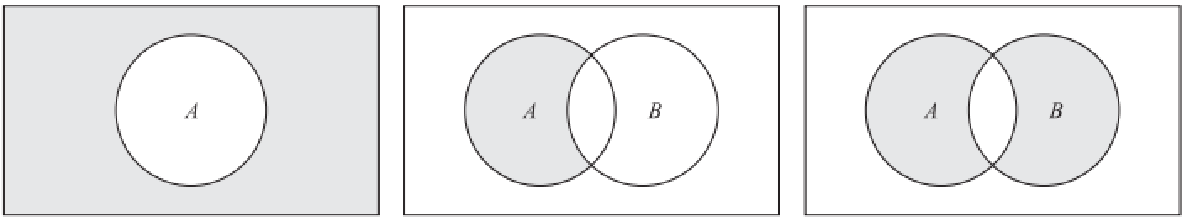
(ii) $A \cap B \subseteq B \subseteq A \cup B$.

1.4.2 Complements, Differences, Symmetric Differences

Recall that all sets under consideration at a particular time are subsets of a fixed universal set U. The absolute complement or, simply, complement of a set A, denoted by A^C, is the set of elements which belong to U but which do not belong to A. That is,

$A^C = \{x \mid x \in U, x \notin A\}$

Some texts denote the complement of A by A' or $\bar{A}$. Fig. 1-3(a) is a Venn diagram in which AC is shaded.



(a) A^C is shaded (b) $A \backslash B$ is shaded (c) $A \oplus B$ is shaded

Fig. 1-3

The relative complement of a set B with respect to a set A or, simply, the difference of A and B, denoted by A\B, is the set of elements which belong to A but which do not belong to B; that is

$A \backslash B = \{x \mid x \in A, x \notin B\}$

The set A\B is read "A minus B." Many texts denote A\B by A − B or A ∼ B. Fig. 1-3(b) is a Venn diagram in which A\B is shaded.

The symmetric difference of sets A and B, denoted by $A \oplus B$, consists of those elements which belong to A or B but not to both. That is, A ⊕ B = (A ∪ B)\(A ∩ B) or A ⊕ B = (A\B) ∪ (B\A)

Figure 1-3(c) is a Venn diagram in which A ⊕ B is shaded.

EXAMPLE 1.4 Suppose U = N = {1, 2, 3,...} is the universal set. Let

A = {1, 2, 3, 4}, B = {3, 4, 5, 6, 7}, C = {2, 3, 8, 9}, E = {2, 4, 6,...}

(Here E is the set of even integers.) Then:

$A^C = \{5, 6, 7, ...\}$, $B^C = \{1, 2, 8, 9, 10, ...\}$, $E^C = \{1, 3, 5, 7, ...\}$

That is, E^C is the set of odd positive integers. Also:

A\B = {1, 2}, A\C = {1, 4}, B\C = {4, 5, 6, 7}, A\E = {1, 3},

B\A = {5, 6, 7}, C\A = {8, 9}, C\B = {2, 8, 9}, E\A = {6, 8, 10, 12,...}.

Furthermore:

A ⊕ B = (A\B) ∪ (B\A) = {1, 2, 5, 6, 7}, B ⊕ C = {2, 4, 5, 6, 7, 8, 9},

A ⊕ C = (A\C) ∪ (B\C) = {1, 4, 8, 9}, A ⊕ E = {1, 3, 6, 8, 10,...}.

1.5 BOOLEAN ALGEBRA OF SETS, DUALITY

Sets under the operations of union, intersection, and complement satisfy various laws (identities) which are listed in Table 1-1. In fact, we formally state this as:

Theorem 1.4: Sets satisfy the laws in Table 1-1.

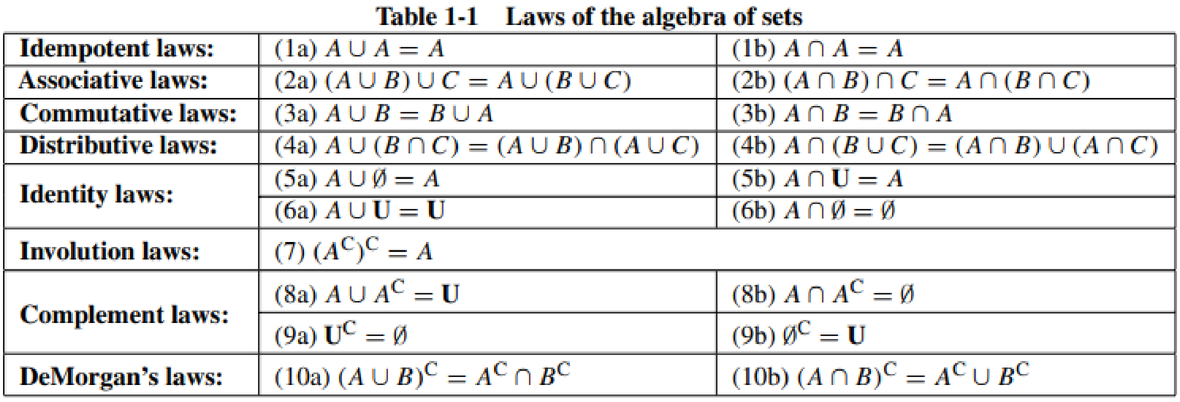


Table 1-1 Laws of the algebra of sets

Idempotent laws:	(1a) $A \cup A = A$	(1b) $A \cap A = A$
Associative laws:	(2a) $(A \cup B) \cup C = A \cup (B \cup C)$	(2b) $(A \cap B) \cap C = A \cap (B \cap C)$
Commutative laws:	(3a) $A \cup B = B \cup A$	(3b) $A \cap B = B \cap A$
Distributive laws:	(4a) $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$	(4b) $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
Identity laws:	(5a) $A \cup \emptyset = A$	(5b) $A \cap \mathbf{U} = A$
	(6a) $A \cup \mathbf{U} = \mathbf{U}$	(6b) $A \cap \emptyset = \emptyset$
Involution laws:	(7) $(A^C)^C = A$	
Complement laws:	(8a) $A \cup A^C = \mathbf{U}$	(8b) $A \cap A^C = \emptyset$
	(9a) $\mathbf{U}^C = \emptyset$	(9b) $\emptyset^C = \mathbf{U}$
DeMorgan's laws:	(10a) $(A \cup B)^C = A^C \cap B^C$	(10b) $(A \cap B)^C = A^C \cup B^C$

Remark: Each law in Table 1-1 follows from an equivalent logical law. Consider, for example, the proof of DeMorgan's Law 10(a):

$(A \cup B)^C = \{x \mid x \notin (A \text{ or } B)\}=\{x \mid x \notin A \text{ and } x \notin B\} = A^C \cap B^C$

Here we use the equivalent (DeMorgan's) logical law:

$\neg(p \vee q) = \neg p \wedge \neg q$

where ¬ means “not,” ∨ means “or,” and ∧ means “and.” (Sometimes Venn diagrams are used to illustrate the laws in Table 1-1.

1.5.1 Duality

The identities in Table 1-1 are arranged in pairs, as, for example, (2a) and (2b). We now consider the principle behind this arrangement. Suppose E is an equation of set algebra. The dual E^* of E is the equation obtained by replacing each occurrence of ∪, ∩, **U** and ∅ in E by ∩, ∪, ∅, and **U**, respectively. For example, the dual of

$$(\mathbf{U} \cap A) \cup (B \cap A) = A \text{ is } (\emptyset \cup A) \cap (B \cup A) = A$$

Observe that the pairs of laws in Table 1-1 are duals of each other. It is a fact of set algebra, called the principle of duality, that if any equation E is an identity then its dual E^* is also an identity.

1.6 FINITE SETS, COUNTING PRINCIPLE

Sets can be finite or infinite. A set S is said to be finite if S is empty or if S contains exactly m elements where m is a positive integer; otherwise S is infinite.

EXAMPLE 1.5

(a) The set A of the letters of the English alphabet and the set D of the days of the week are finite sets. Specifically,

A has 26 elements and D has 7 elements.

(b) Let E be the set of even positive integers, and let I be the unit interval, that is,

$$E = \{2, 4, 6, ...\} \text{ and } I = [0, 1]=\{x \mid 0 \le x \le 1\}$$

Then both E and I are infinite.

A set S is countable if S is finite or if the elements of S can be arranged as a sequence, in which case S is said to be countably infinite; otherwise S is said to be uncountable. The above set E of even integers is countably infinite, whereas one can prove that the unit interval I = [0, 1] is uncountable.

1.6.1 Counting Elements in Finite Sets

The notation n(S) or |S| will denote the number of elements in a set S. (Some texts use #(S) or card(S) instead of n(S).) Thus n(A) = 26, where A is the letters in the English alphabet, and n(D) = 7, where D is the days of the week. Also n(∅) = 0 since the empty set has no elements.

The following lemma applies.

Lemma 1.1: Suppose A and B are finite disjoint sets. Then A ∪ B is finite and

$$n(A \cup B) = n(A) + n(B)$$

Proof. In counting the elements of A ∪ B, first count those that are in A. There are n(A) of these. The only other elements of A ∪ B are those that are in B but not in A. But since A and B are disjoint, no element of B is in A, so there are n(B) elements that are in B but not in A. Therefore, n(A ∪ B) = n(A) + n(B).

For any sets A and B, the set A is the disjoint union of A\B and A ∩ B. Thus Lemma 1.1 gives us the following useful result.

Corollary 1.1: Let A and B be finite sets. Then

$$n(A\backslash B) = n(A) - n(A \cap B)$$

For example, suppose an art class A has 25 students and 10 of them are taking a biology class B. Then the number of students in class A which are not in class B is:

$$n(A\backslash B) = n(A) - n(A \cap B) = 25 - 10 = 15$$

Given any set A, recall that the universal set **U** is the disjoint union of A and A^C. Accordingly, Lemma 1.1 also gives the following result.

Corollary 1.2: Let A be a subset of a finite universal set **U**. Then

$$n(A^C) = n(\mathbf{U}) - n(A)$$

For example, suppose a class **U** with 30 students has 18 full-time students. Then there are 30−18 = 12 part-time students in the class **U**.

1.6.2 Inclusion–Exclusion Principle

There is a formula for n(A ∪ B) even when they are not disjoint, called the Inclusion–Exclusion Principle. Namely:

Theorem 1.5: Suppose A and B are finite sets. Then A ∪ B and A ∩ B are finite and

$$n(A \cup B) = n(A) + n(B) - n(A \cap B)$$

That is, we find the number of elements in A or B (or both) by first adding n(A) and n(B) (inclusion) and then subtracting n(A ∩ B) (exclusion) since its elements were counted twice.

We can apply this result to obtain a similar formula for three sets:

Corollary 1.3: Suppose A, B, C are finite sets. Then A ∪ B ∪ C is finite and

$$n(A \cup B \cup C) = n(A) + n(B) + n(C) - n(A \cap B) - n(A \cap C) - n(B \cap C) + n(A \cap B \cap C)$$

EXAMPLE 1.6: Suppose a list A contains the 30 students in a mathematics class, and a list B contains the 35 students in an English class, and suppose there are 20 names on both lists. Find the number of students: (a) only on list A, (b) only on list B, (c) on list A or B (or both), (d) on exactly one list.

(a) List A has 30 names and 20 are on list B; hence 30 − 20 = 10 names are only on list A.

(b) Similarly, 35 − 20 = 15 are only on list B.

(c) We seek n(A ∪ B). By inclusion–exclusion,

$$n(A \cup B) = n(A) + n(B) - n(A \cap B) = 30 + 35 - 20 = 45.$$

In other words, we combine the two lists and then cross out the 20 names which appear twice.

(d) By (a) and (b), 10 + 15 = 25 names are only on one list; that is, $n(A \oplus B) = 25$.

1.7 CLASSES OF SETS, POWER SETS, PARTITIONS

Given a set S, we might wish to talk about some of its subsets. Thus we would be considering a set of sets. Whenever such a situation occurs, to avoid confusion, we will speak of a class of sets or collection of sets rather than a set of sets. If we wish to consider some of the sets in a given class of sets, then we speak of subclass or subcollection.

EXAMPLE 1.7 Suppose S = {1, 2, 3, 4}.

(a) Let A be the class of subsets of S which contain exactly three elements of S. Then

$$A = [\{1, 2, 3\},\{1, 2, 4\},\{1, 3, 4\},\{2, 3, 4\}]$$

That is, the elements of A are the sets {1, 2, 3}, {1, 2, 4}, {1, 3, 4}, and {2, 3, 4}.

(b) Let B be the class of subsets of S, each which contains 2 and two other elements of S. Then

$$B = [\{1, 2, 3\},\{1, 2, 4\},\{2, 3, 4\}]$$

The elements of B are the sets {1, 2, 3}, {1, 2, 4}, and {2, 3, 4}. Thus B is a subclass of A, since every element of B is also an element of A. (To avoid confusion, we will sometimes enclose the sets of a class in brackets instead of braces.)

1.7.1 Power Sets

For a given set S, we may speak of the class of all subsets of S. This class is called the power set of S, and will be denoted by P(S). If S is finite, then so is P(S). In fact, the number of elements in P(S) is 2 raised to the power n(S). That is,

$$n(P\ (S)) = 2^{n(S)}$$

(For this reason, the power set of S is sometimes denoted by 2^S.)

EXAMPLE 1.8 Suppose S = {1, 2, 3}. Then

$$P\ (S) = [\emptyset,\{1\},\{2\},\{3\},\{1, 2\},\{1, 3\},\{2, 3\}, S]$$

Note that the empty set ∅ belongs to P(S) since ∅ is a subset of S. Similarly, S belongs to P(S). As expected from the above remark, P(S) has $2^3 = 8$ elements.

1.7.2 Partitions

Let S be a nonempty set. A partition of S is a subdivision of S into nonoverlapping, nonempty subsets. Precisely, a partition of S is a collection {Ai} of nonempty subsets of S such that:

(i) Each a in S belongs to one of the Ai.

(ii) The sets of {Ai} are mutually disjoint; that is, if

$$A_j \neq A_k \text{ then } A_j \cap A_k = \emptyset$$

The subsets in a partition are called cells. Figure 1-6 is a Venn diagram of a partition of the rectangular set S of points into five cells, A_1, A_2, A_3, A_4, A_5.

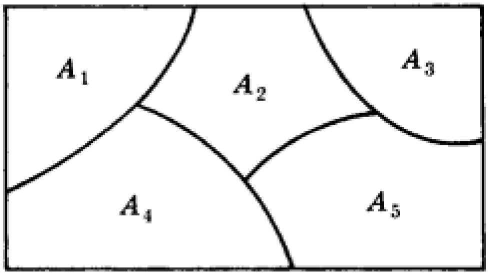


Fig. 1-4

EXAMPLE 1.9 Consider the following collections of subsets of S = {1, 2, . . ., 8, 9}:

(i) [{1, 3, 5}, {2, 6}, {4, 8, 9}]

(ii) [{1, 3, 5}, {2, 4, 6, 8}, {5, 7, 9}]

(iii) [{1, 3, 5}, {2, 4, 6, 8}, {7, 9}]

Then (i) is not a partition of S since 7 in S does not belong to any of the subsets. Furthermore, (ii) is not a partition of S since {1, 3, 5} and {5, 7, 9} are not disjoint. On the other hand, (iii) is a partition of S.

2.1 RELATIONS

The student is familiar with many relations such as "less than," "is parallel to," "is a subset of," and so on. In a certain sense, these relations consider the existence or nonexistence of a certain connection between pairs of objects taken in a definite order. Formally, we define a relation in terms of these "ordered pairs."

An ordered pair of elements a and b, where a is designated as the first element and b as the second element, is denoted by (a, b). In particular,

$$(a, b) = (c, d)$$

if and only if a = c and b = d. Thus (a, b) ≠ (b, a) unless a = b. This contrasts with sets where the order of elements is irrelevant; for example, {3, 5}={5, 3}.

2.2 PRODUCT SETS

Consider two arbitrary sets A and B. The set of all ordered pairs (a, b) where a ∈ A and b ∈ B is called the product, or Cartesian product, of A and B. A short designation of this product is A × B, which is read "A cross B." By definition,

$$A \times B = \{(a, b) | a \in A \text{ and } b \in B\}$$

One frequently writes A2 instead of A × A.

EXAMPLE 2.1 R denotes the set of real numbers and so R^2 = R×R is the set of ordered pairs of real numbers. The student is familiar with the geometrical representation of R^2 as points in the plane as in Fig. 2-1. Here each point P represents an ordered pair (a, b) of real numbers and vice versa; the vertical line through P meets the x-axis at a, and the horizontal line through P meets the y-axis at b. R^2 is frequently called the Cartesian plane.

EXAMPLE 2.2 Let $A = \{1, 2\}$ and $B = \{a, b, c\}$. Then

$$A \times B = \{(1, a), (1, b), (1, c), (2, a), (2, b), (2, c)\}$$

$$B \times A = \{(a, 1), (b, 1), (c, 1), (a, 2), (b, 2), (c, 2)\}$$

Also, $A \times A = \{(1, 1), (1, 2), (2, 1), (2, 2)\}$

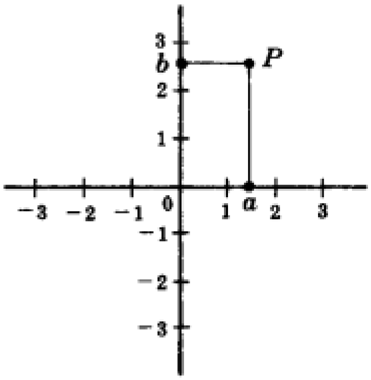


Fig. 2-1

There are two things worth noting in the above examples. First of all $A \times B \neq B \times A$. The Cartesian product deals with ordered pairs, so naturally the order in which the sets are considered is important. Secondly, using $n(S)$ for the number of elements in a set S, we have:

$$n(A \times B) = 6 = 2(3) = n(A)n(B)$$

In fact, $n(A \times B) = n(A)n(B)$ for any finite sets A and B. This follows from the observation that, for an ordered pair (a, b) in $A \times B$, there are $n(A)$ possibilities for a, and for each of these there are $n(B)$ possibilities for b.

The idea of a product of sets can be extended to any finite number of sets. Just as we write A^2 instead of $A \times A$, so we write A^n instead of $A \times A \times \cdots \times A$, where there are n factors all equal to A. For example, $R^3 = R \times R \times R$ denotes the usual three-dimensional space.

2.3 RELATIONS

We begin with a definition.

Definition 2.1: Let A and B be sets. A binary relation or, simply, relation from A to B is a subset of $A \times B$.

Suppose R is a relation from A to B. Then R is a set of ordered pairs where each first element comes from A and each second element comes from B. That is, for each pair $a \in A$ and $b \in B$, exactly one of the following is true:

(i) $(a, b) \in R$; we then say "a is R-related to b", written aRb.

(ii) $(a, b) \notin R$; we then say "a is not R-related to b", written $a \not{R} b$.

If R is a relation from a set A to itself, that is, if R is a subset of $A^2 = A \times A$, then we say that R is a relation on A.

The domain of a relation R is the set of all first elements of the ordered pairs which belong to R, and the range is the set of second elements.

EXAMPLE 2.3

(a) A = (1, 2, 3) and B = {x, y, z}, and let R = {(1, y), (1, z), (3, y)}. Then R is a relation from A to B since R is a subset of A × B. With respect to this relation,

$$1Ry,\ 1Rz,\ 3Ry, \quad \text{but} \quad 1\not{R}x,\ 2\not{R}x,\ 2\not{R}y,\ 2\not{R}z,\ 3\not{R}x,\ 3\not{R}z$$

The domain of R is {1, 3} and the range is {y, z}.

(b) Set inclusion ⊆ is a relation on any collection of sets. For, given any pair of set A and B, either A ⊆ B or A ⊈ B.

(c) A familiar relation on the set Z of integers is “m divides n.” A common notation for this relation is to write m|n when m divides n. Thus 6 | 30 but 7 ∤ 25.

(d) Consider the set L of lines in the plane. Perpendicularity, written “⊥,” is a relation on L. That is, given any pair of lines a and b, either a ⊥ b or $a \not\perp b$.

Similarly, “is parallel to,” written “||,” is a relation on L since either a || b or a ∦ b.

(e) Let A be any set. An important relation on A is that of equality,

$\{(a, a) \mid a \in A\}$

which is usually denoted by “=.” This relation is also called the identity or diagonal relation on A and it will also be denoted by $\triangle_A$ or simply △.

(f) Let A be any set. Then A × A and ∅ are subsets of A × A and hence are relations on A called the universal relation and empty relation, respectively.

2.3.1 Inverse Relation

Let R be any relation from a set A to a set B. The inverse of R, denoted by R^{-1}, is the relation from B to A which consists of those ordered pairs which, when reversed, belong to R; that is,

$R^{-1} = \{(b, a) \mid (a, b) \in R\}$

For example, let A = {1, 2, 3} and B = {x, y,z}. Then the inverse of

R = {(1, y), (1, z), (3, y)} is R^{-1} = {(y, 1), (z, 1), (y, 3)}

Clearly, if R is any relation, then $(R^{-1})^{-1} = R$. Also, the domain and range of R^{-1} are equal, respectively, to the range and domain of R. Moreover, if R is a relation on A, then R^{-1} is also a relation on A.

2.4 PICTORIAL REPRESENTATIVES OF RELATIONS

There are various ways of picturing relations.

2.4.1 Relations on R

Let S be a relation on the set R of real numbers; that is, S is a subset of $R^2 = R \times R$. Frequently, S consists of all ordered pairs of real numbers which satisfy some given equation $E(x, y) = 0$ (such as $x^2 + y^2 = 25$).

Since R^2 can be represented by the set of points in the plane, we can picture S by emphasizing those points in the plane which belong to S. The pictorial representation of the relation is sometimes called the graph of the relation. For example, the graph of the relation $x^2 + y^2 = 25$ is a circle having its center at the origin and radius 5. See Fig. 2-2(a).

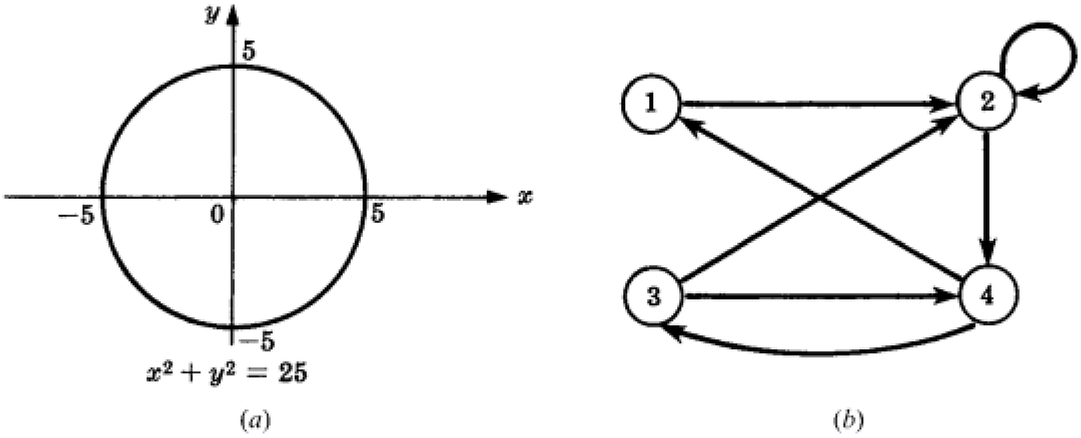


Fig. 2-2

2.4.2 Directed Graphs of Relations on Sets

There is an important way of picturing a relation R on a finite set. First we write down the elements of the set, and then we draw an arrow from each element x to each element y whenever x is related to y. This diagram is called the directed graph of the relation. Figure 2-2(b), for example, shows the directed graph of the following relation R on the set A = {1, 2, 3, 4}:

$$R = \{(1, 2), (2, 2), (2, 4), (3, 2), (3, 4), (4, 1), (4, 3)\}$$

Observe that there is an arrow from 2 to itself, since 2 is related to 2 under R.

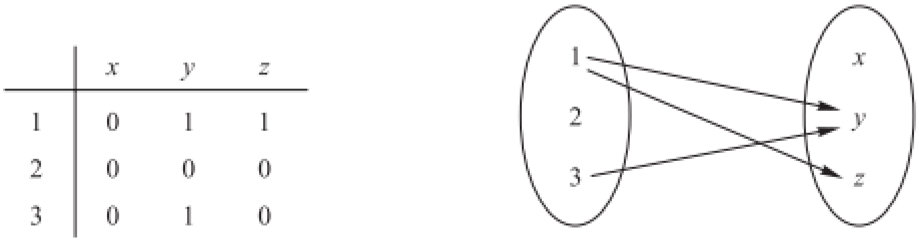
2.4.3 Pictures of Relations on Finite Sets

Suppose A and B are finite sets. There are two ways of picturing a relation R from A to B.

(i) Form a rectangular array (matrix) whose rows are labeled by the elements of A and whose columns are labeled by the elements of B. Put a 1 or 0 in each position of the array according as $a \in A$ is or is not related to $b \in B$. This array is called the matrix of the relation.

(ii) Write down the elements of A and the elements of B in two disjoint disks, and then draw an arrow from $a \in A$ to $b \in B$ whenever a is related to b. This picture will be called the arrow diagram of the relation.

Figure 2-3 pictures the relation R in Example 2.3(a) by the above two ways.



	x	y	z
1	0	1	1
2	0	0	0
3	0	1	0

(i) (ii)

$R = \{(1, y),\ (1, z),\ (3, y)\}$

Fig. 2-3

2.5 COMPOSITION OF RELATIONS

Let A, B and C be sets, and let R be a relation from A to B and let S be a relation from B to C. That is, R is a subset of A × B and S is a subset of B × C. Then R and S give rise to a relation from A to C denoted by R◦S and defined by:

a(R◦S)c if for some b ∈ B we have aRb and bSc.

That is ,

R ◦ S = {(a, c)| there exists b ∈ B for which (a, b) ∈ R and (b, c) ∈ S}

The relation R◦S is called the composition of R and S; it is sometimes denoted simply by RS.

Suppose R is a relation on a set A, that is, R is a relation from a set A to itself. Then R◦R, the composition of R with itself, is always defined. Also, R◦R is sometimes denoted by R^2. Similarly, $R^3 = R^2 \circ R = R \circ R \circ R$, and so on. Thus R^n is defined for all positive n.

EXAMPLE 2.4 Let A = {1, 2, 3, 4}, B = {a, b, c, d}, C = {x, y, z} and let

R = {(1, a), (2, d), (3, a), (3, b), (3,d)} and S = {(b, x), (b, z), (c, y), (d, z)}

Consider the arrow diagrams of R and S as in Fig. 2-4. Observe that there is an arrow from 2 to d which is followed by an arrow from d to z. We can view these two arrows as a “path” which “connects” the element 2 ∈ A to the element z ∈ C. Thus:

2(R ◦ S)z since 2Rd and dSz

Similarly there is a path from 3 to x and a path from 3 to z. Hence

3(R◦S)x and 3(R◦S)z

No other element of A is connected to an element of C. Accordingly,

R ◦ S = {(2, z), (3, x), (3, z)}

Our first theorem tells us that composition of relations is associative.

Theorem 2.1: Let A, B, C and D be sets. Suppose R is a relation from A to B, S is a relation from B to C, and T is a relation from C to D. Then

(R ◦ S) ◦ T = R ◦ (S ◦ T)

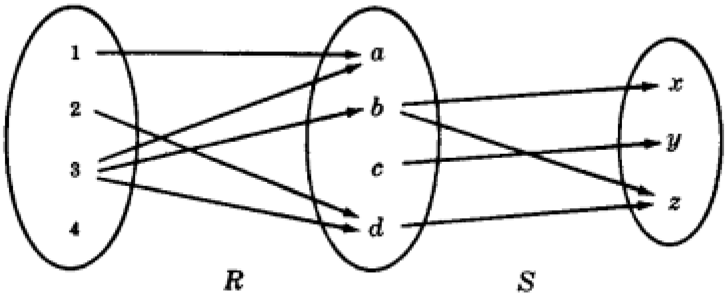


Fig. 2-4

2.5.1 Composition of Relations and Matrices

There is another way of finding R◦S. Let MR and MS denote respectively the matrix representations of the relations R and S. Then

$$M_R = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \quad \text{and} \quad M_S = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{ccc} x & y & z \end{array} \\ \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$

Multiplying MR and MS we obtain the matrix

$$M = M_R M_S = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccc} x & y & z \end{array} \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 2 \\ 0 & 0 & 0 \end{bmatrix} \end{array}$$

The nonzero entries in this matrix tell us which elements are related by R◦S. Thus M = MRMS and MR◦S have the same nonzero entries.

2.6 TYPES OF RELATIONS

This unit discusses important types of relations defined on a set A.

2.6.1 Reflexive Relations

A relation R on a set A is reflexive if aRa for every a ∈ A, that is, if (a, a) ∈ R for every a ∈ A. Thus R is not reflexive if there exists a ∈ A such that (a, a) ∉ R.

EXAMPLE 2.5 Consider the following five relations on the set A = {1, 2, 3, 4}:

R_1 = {(1, 1), (1, 2), (2, 3), (1, 3), (4, 4)}

R_2 = {(1, 1)(1, 2), (2, 1), (2, 2), (3, 3), (4, 4)}

R_3 = {(1, 3), (2, 1)}

R_4 = ∅, the empty relation

$R_5 = A \times A$, the universal relation

Determine which of the relations are reflexive.

Since A contains the four elements 1, 2, 3, and 4, a relation R on A is reflexive if it contains the four pairs (1, 1), (2, 2), (3, 3), and (4, 4). Thus only R_2 and the universal relation $R_5 = A \times A$ are reflexive. Note that R_1, R_3, and R_4 are not reflexive since, for example, (2, 2) does not belong to any of them.

EXAMPLE 2.6 Consider the following five relations:

(1) Relation ≤ (less than or equal) on the set **Z** of integers.

(2) Set inclusion ⊆ on a collection **C** of sets.

(3) Relation ⊥ (perpendicular) on the set L of lines in the plane.

(4) Relation ǁ (parallel) on the set L of lines in the plane.

(5) Relation | of divisibility on the set **N** of positive integers. (Recall $x \mid y$ if there exists z such that $xz = y$.)

Determine which of the relations are reflexive.

The relation (3) is not reflexive since no line is perpendicular to itself. Also (4) is not reflexive since no line is parallel to itself. The other relations are reflexive; that is, $x \leq x$ for every $x \in Z$, $A \subseteq A$ for any set $A \in C$, and $n|n$ for every positive integer $n \in$ **N**.

2.6.2 Symmetric and Antisymmetric Relations

A relation R on a set A is symmetric if whenever aRb then bRa, that is, if whenever(a, b) ∈ R then (b, a) ∈ R.

Thus R is not symmetric if there exists $a, b \in A$ such that $(a, b) \in R$ but $(b, a) \notin R$.

EXAMPLE 2.7

(a) Determine which of the relations in Example 2.5 are symmetric.

R1 is not symmetric since (1, 2) ∈ R1 but (2, 1) ∉ R1. R3 is not symmetric since (1, 3) ∈ R3 but (3, 1) ∉ R3.

The other relations are symmetric.

(b) Determine which of the relations in Example 2.6 are symmetric.

The relation ⊥ is symmetric since if line a is perpendicular to line b then b is perpendicular to a. Also, ǁ is symmetric since if line a is parallel to line b then b is parallel to line a. The other relations are not symmetric. For example:

$3 \leq 4$ but $4 \nleq 3$; $\{1, 2\} \subseteq \{1, 2, 3\}$ but $\{1, 2, 3\} \nsubseteq \{1, 2\}$; and $2 \mid 6$ but $6 \nmid 2$.

A relation R on a set A is antisymmetric if whenever aRb and bRa then a = b, that is, if $a \neq b$ and aRb then $b \not{R} a$. Thus R is not antisymmetric if there exist distinct elements a and b in A such that aRb and bRa.

EXAMPLE 2.8

(a) Determine which of the relations in Example 2.5 are antisymmetric.

R_2 is not antisymmetric since (1, 2) and (2, 1) belong to R_2, but 1 ≠2. Similarly, the universal relation R_3 is not antisymmetric. All the other relations are antisymmetric.

(b) Determine which of the relations in Example 2.6 are antisymmetric.

The relation ≤ is antisymmetric since whenever a ≤ b and b ≤ a then a = b. Set inclusion ⊆ is antisymmetric since whenever A ⊆ B and B ⊆ A then A = B. Also, divisibility on **N** is antisymmetric since whenever m | n and n | m then m = n. (Note that divisibility on Z is not antisymmetric since 3 | −3 and −3 | 3 but 3 ≠ −3.) The relations ⊥ and ∥ are not antisymmetric.

2.6.3 Transitive Relations

A relation R on a set A is transitive if whenever aRb and bRc then aRc, that is, if whenever (a, b), (b, c) ∈ R then (a, c) ∈ R. Thus R is not transitive if there exist a, b, c ∈ R such that (a, b), (b, c) ∈ R but (a, c) ∉ R.

EXAMPLE 2.9

(a) Determine which of the relations in Example 2.5 are transitive.

The relation R_3 is not transitive since (2, 1),(1, 3) ∈ R_3 but (2, 3) ∉ R_3. All the other relations are transitive.

(b) Determine which of the relations in Example 2.6 are transitive.

2.7 CLOSURE PROPERTIES

Let R be a relation defined on a set A, and if P is a set of properties, then the property closure of a relation R, denoted as P-closure, is the smallest relation, R′, which has the properties mentioned in P. It is obtained by adding every pair (a, b) in R to R′, and then adding those pairs of the members of A that will make relation R have the properties in P. If P contains only transitivity properties, then the P-closure will be called as a transitive closure of the relation, and we denote the transitive closure of relation R by R^+; whereas when P contains transitive as well as reflexive properties, then the P-closure is called as a reflexive-transitive closure of relation R, and we denote it by R^*.

For example, if:

R = { (0, 1), (1, 2), (3, 4) } then

R^+ ={ (0, 1), (1, 2), (3, 4), (0, 2) }

R^* ={ (0, 1), (1, 2), (3, 4), (0, 2), (0, 0), (1, 1), (2, 2), (3, 3), (4, 4) }

$R^* = R^+ \cup \{(a, a)$ | for every a in A }

2.8 EQUIVALENCE RELATIONS

Consider a nonempty set S. A relation R on S is an equivalence relation if R is reflexive, symmetric, and transitive. That is, R is an equivalence relation on S if it has the following three properties:

(1) For every a ∈ S, aRa. (2) If aRb, then bRa. (3) If aRb and bRc, then aRc

The general idea behind an equivalence relation is that it is a classification of objects which are in some way "alike." In fact, the relation "=" of equality on any set S is an equivalence relation; that is:

(1) a = a for every a ∈ S. (2) If a = b, then b = a. (3) If a = b, b = c, then a = c.

3.1 LOGIC

3.2 INTRODUCTION

Many algorithms and proofs use logical expressions such as:

"IF p THEN q" or "If p1 AND p2, THEN q1 OR q2"

Therefore, it is necessary to know the cases in which these expressions are TRUE or FALSE, that is, to know the "truth value" of such expressions. We discuss these issues in this lesson.

We also investigate the truth value of quantified statements, which are statements which use the logical quantifiers "for every" and "there exist."

3.3 PROPOSITIONS AND COMPOUND STATEMENTS

A proposition (or statement) is a declarative statement which is true or false, but not both. Consider, for example, the following six sentences:

(i) Ice floats in water. (iii) 2 + 2 = 4 (v) Where are you going?

(ii) China is in Europe. (iv) 2 + 2 = 5 (vi) Do your homework.

The first four are propositions, the last two are not. Also, (i) and (iii) are true, but (ii) and (iv) are false.

3.3.1 Compound Propositions

Many propositions are composite, that is, composed of subpropositions and various connectives discussed subsequently. Such composite propositions are called compound propositions. A proposition is said to be primitive if it cannot be broken down into simpler propositions, that is, if it is not composite.

For example, the above propositions (i) through (iv) are primitive propositions. On the other hand, the following two propositions are composite:

"Roses are red and violets are blue." and "John is smart or he studies every night."

The fundamental property of a compound proposition is that its truth value is completely determined by the truth values of its subpropositions together with the way in which they are connected to form the compound propositions.

3.4 BASIC LOGICAL OPERATIONS

There are three basic logical operations of conjunction, disjunction, and negation which correspond, respectively, to the English words “and,” “or,” and “not.”

3.4.1 Conjunction, $p \wedge q$

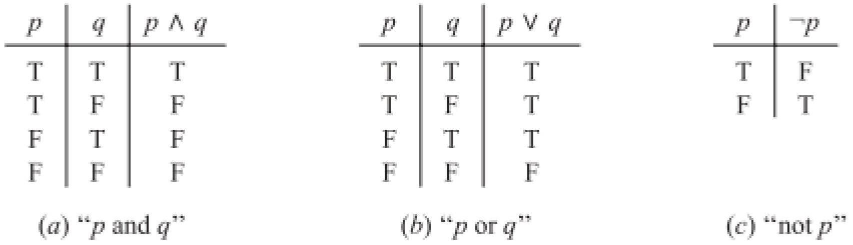
Any two propositions can be combined by the word “and” to form a compound proposition called the conjunction of the original propositions. Symbolically,

$$p \wedge q$$

read “p and q,” denotes the conjunction of p and q. Since p ∧ q is a proposition it has a truth value, and this truth value depends only on the truth values of p and q. Specifically:

Definition 3.1: If p and q are true, then p ∧ q is true; otherwise p ∧ q is false.

The truth value of p ∧ q may be defined equivalently by the table in Fig. 3-1(a). Here, the first line is a short way of saying that if p is true and q is true, then p ∧ q is true. The second line says that if p is true and q is false, then p ∧ q is false. And so on. Observe that there are four lines corresponding to the four possible combinations of T and F for the two subpropositions p and q. Note that p ∧ q is true only when both p and q are true.



p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

(*a*) “*p* and *q*”

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

(*b*) “*p* or *q*”

p	$\neg p$
T	F
F	T

(*c*) “not *p*”

Fig. 3-1.

EXAMPLE 3.1 Consider the following four statements:

(i) Ice floats in water and 2 + 2 = 4. (iii) China is in Europe and 2 + 2 = 4.

(ii) Ice floats in water and 2 + 2 = 5. (iv) China is in Europe and 2 + 2 = 5.

Only the first statement is true. Each of the others is false since at least one of its substatements is false.

3.4.2 Disjunction, p ∨ q

Any two propositions can be combined by the word “or” to form a compound proposition called the disjunction of the original propositions. Symbolically,

$$p \vee q$$

read “p or q,” denotes the disjunction of p and q. The truth value of p ∨ q depends only on the truth values of p and q as follows.

Definition 3.2: If p and q are false, then p ∨ q is false; otherwise p ∨ q is true.

The truth value of $p \vee q$ may be defined equivalently by the table in Fig. 3-1(b). Observe that $p \vee q$ is false only in the fourth case when both p and q are false.

EXAMPLE 3.2 Consider the following four statements:

(i) Ice floats in water or 2 + 2 = 4. (iii) China is in Europe or 2 + 2 = 4.

(ii) Ice floats in water or 2 + 2 = 5. (iv) China is in Europe or 2 + 2 = 5.

Only the last statement (iv) is false. Each of the others is true since at least one of its sub statements is true.

3.4.3 Negation, ¬p

Given any proposition p, another proposition, called the negation of p, can be formed by writing "It is not true that ..." or "It is false that ..." before p or, if possible, by inserting in p the word "not." Symbolically, the negation of p, read "not p," is denoted by

$$\neg p$$

The truth value of $\neg p$ depends on the truth value of p as follows:

Definition 3.3: If p is true, then $\neg p$ is false; and if p is false, then $\neg p$ is true.

The truth value of $\neg p$ may be defined equivalently by the table in Fig. 3-1(c). Thus the truth value of the negation of p is always the opposite of the truth value of p.

EXAMPLE 3.3 Consider the following six statements:

(a1) Ice floats in water. (a2) It is false that ice floats in water. (a3) Ice does not float in water.

(b1) 2 + 2 = 5 (b2) It is false that 2 + 2 = 5. (b3) $2 + 2 \neq 5$

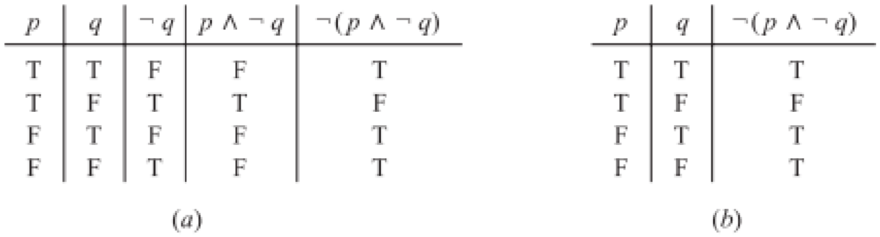
Then (a2) and (a3) are each the negation of (a1); and (b2) and (b3) are each the negation of (b1). Since (a1) is true, (a2) and (a3) are false; and since (b1) is false, (b2) and (b3) are true.

3.5 PROPOSITIONS AND TRUTH TABLES

Let P (p, q, . . .) denote an expression constructed from logical variables p, q,..., which take on the value TRUE (T) or FALSE (F), and the logical connectives $\wedge$, $\vee$, and $\neg$ (and others discussed subsequently). Such an expression P (p, q, . . .) will be called a proposition.

The main property of a proposition P (p, q, . . .) is that its truth value depends exclusively upon the truth values of its variables, that is, the truth value of a proposition is known once the truth value of each of its variables is known. A simple concise way to show this relationship is through a truth table.

Consider, for example, the proposition $\neg(p \wedge \neg q)$. Figure 3-2(a) indicates how the truth table of $\neg(p \wedge \neg q)$ is constructed. The actual truth table of the proposition $\neg(p\wedge\neg q)$ is shown in Fig. 3-2(b). It consists precisely of the columns in Fig. 3-2(a) which appear under the variables and under the proposition; the other columns were merely used in the construction of the truth table.



p	q	$\neg q$	$p \wedge \neg q$	$\neg(p \wedge \neg q)$
T	T	F	F	T
T	F	T	T	F
F	T	F	F	T
F	F	T	F	T

(*a*)

p	q	$\neg(p \wedge \neg q)$
T	T	T
T	F	F
F	T	T
F	F	T

(*b*)

Fig. 3-2.

3.5.1 Order of precedence

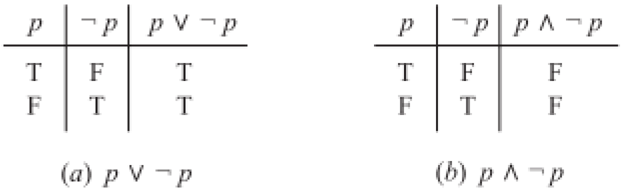
In order to avoid an excessive number of parentheses, we sometimes adopt an order of precedence for the logical connectives. Specifically,

¬ has precedence over ∧ which has precedence over ∨

For example, ¬p ∧ q means (¬p) ∧ q and not ¬(p ∧ q).

3.6 TAUTOLOGIES AND CONTRADICTIONS

Some propositions P (p, q, . . .) contain only T in the last column of their truth tables or, in other words, they are true for any truth values of their variables. Such propositions are called tautologies. Analogously, a proposition P (p, q, . . .) is called a contradiction if it contains only F in the last column of its truth table or, in other words, if it is false for any truth values of its variables. For example, the proposition “p or not p,” that is, p ∨ ¬p, is a tautology, and the proposition “p and not p,” that is, p ∧ ¬p, is a contradiction. This is verified by looking at their truth tables in Fig. 3-3. (The truth tables have only two rows since each proposition has only the one variable p.)



p	$\neg p$	$p \vee \neg p$
T	F	T
F	T	T

(*a*) $p \vee \neg p$

p	$\neg p$	$p \wedge \neg p$
T	F	F
F	T	F

(*b*) $p \wedge \neg p$

Fig. 3-3.

Note that the negation of a tautology is a contradiction since it is always false, and the negation of a contradiction is a tautology since it is always true.

3.7 LOGICAL EQUIVALENCE

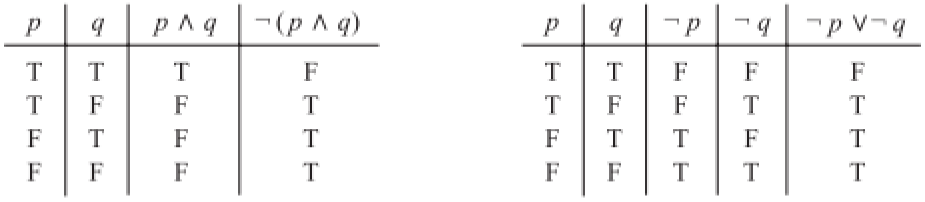
Two propositions P (p, q, . . .) and Q(p, q, . . .) are said to be logically equivalent, or simply equivalent or equal, denoted by

P (p, q, . . .) ≡ Q(p, q, . . .)

if they have identical truth tables. Consider, for example, the truth tables of ¬(p ∧ q) and ¬p ∨ ¬q appearing in Fig. 3-4. Observe that both truth tables are the same, that is, both propositions are false in the first case and true in the other three cases. Accordingly, we can write

¬(p ∧ q) ≡ ¬p ∨ ¬q

In other words, the propositions are logically equivalent.



p	q	$p \wedge q$	$\neg(p \wedge q)$
T	T	T	F
T	F	F	T
F	T	F	T
F	F	F	T

(a) $\neg(p \wedge q)$

p	q	$\neg p$	$\neg q$	$\neg p \vee \neg q$
T	T	F	F	F
T	F	F	T	T
F	T	T	F	T
F	F	T	T	T

(b) $\neg p \vee \neg q$

Fig. 3-4.

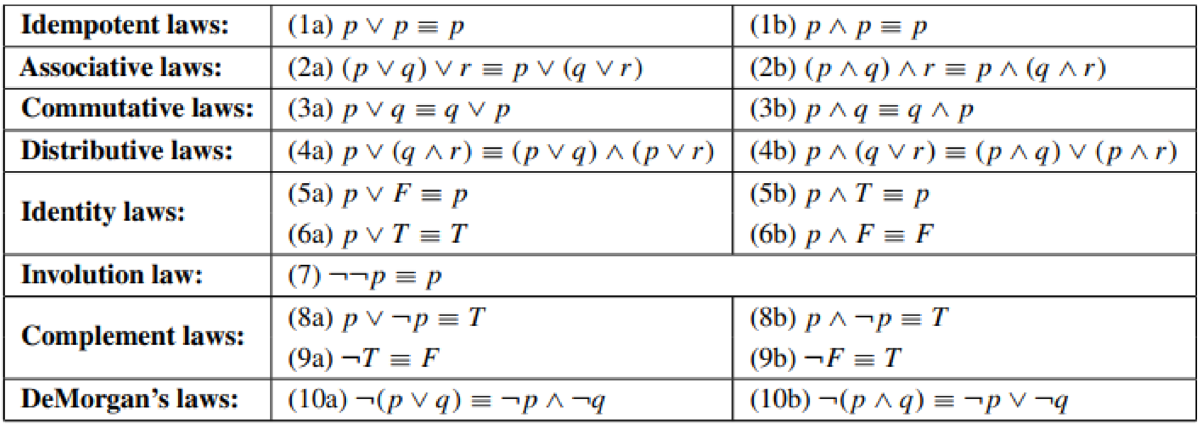
3.8 ALGEBRA OF PROPOSITIONS

Propositions satisfy various laws which are listed in Table 3-1. (In this table, T and F are restricted to the truth values “True” and “False,” respectively.) We state this result formally.

Theorem 3.1: Propositions satisfy the laws of Table 3-1.

(Observe the similarity between this Table 3-1 and Table 1-1 on sets.)

Table 3-1 Laws of the algebra of propositions



Idempotent laws:	(1a) $p \vee p \equiv p$	(1b) $p \wedge p \equiv p$
Associative laws:	(2a) $(p \vee q) \vee r \equiv p \vee (q \vee r)$	(2b) $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
Commutative laws:	(3a) $p \vee q \equiv q \vee p$	(3b) $p \wedge q \equiv q \wedge p$
Distributive laws:	(4a) $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$	(4b) $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
Identity laws:	(5a) $p \vee F \equiv p$ (6a) $p \vee T \equiv T$	(5b) $p \wedge T \equiv p$ (6b) $p \wedge F \equiv F$
Involution law:	(7) $\neg\neg p \equiv p$	
Complement laws:	(8a) $p \vee \neg p \equiv T$ (9a) $\neg T \equiv F$	(8b) $p \wedge \neg p \equiv T$ (9b) $\neg F \equiv T$
DeMorgan’s laws:	(10a) $\neg(p \vee q) \equiv \neg p \wedge \neg q$	(10b) $\neg(p \wedge q) \equiv \neg p \vee \neg q$

3.9 CONDITIONAL AND BICONDITIONAL STATEMENTS

Many statements, particularly in mathematics, are of the form “If p then q.” Such statements are called conditional statements and are denoted by

$$p \rightarrow q$$

The conditional p → q is frequently read “p implies q” or “p only if q.”

Another common statement is of the form “p if and only if q.” Such statements are called biconditional statements and are denoted by

$$p \leftrightarrow q$$

The truth values of p → q and p ↔ q are defined by the tables in Fig. 3-5(a) and (b). Observe that:

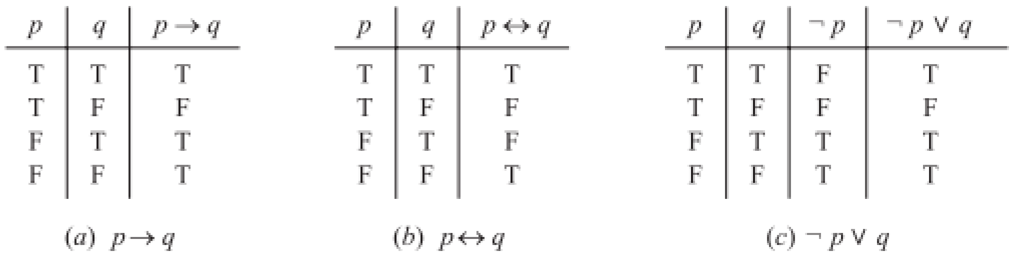
(a) The conditional $p \to q$ is false only when the first part p is true and the second part q is false. Accordingly, when p is false, the conditional $p \to q$ is true regardless of the truth value of q.

(b) The biconditional $p \leftrightarrow q$ is true whenever p and q have the same truth values and false otherwise.

The truth table of $\neg p \land q$ appears in Fig. 3-5(c). Note that the truth table of $\neg p \lor q$ and $p \to q$ are identical, that is, they are both false only in the second case. Accordingly, $p \to q$ is logically equivalent to $\neg p \lor q$; that is,

$$p \to q \equiv \neg p \lor q$$

In other words, the conditional statement "If p then q" is logically equivalent to the statement "Not p or q" which only involves the connectives $\lor$ and $\neg$ and thus was already a part of our language. We may regard $p \to q$ as an abbreviation for an oft-recurring statement.



p	q	$p \to q$
T	T	T
T	F	F
F	T	T
F	F	T

(a) $p \to q$

p	q	$p \leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

(b) $p \leftrightarrow q$

p	q	$\neg p$	$\neg p \lor q$
T	T	F	T
T	F	F	F
F	T	T	T
F	F	T	T

(c) $\neg p \lor q$

Fig. 3-5

3.10 ARGUMENTS

An argument is an assertion that a given set of propositions P1, P2,...,Pn, called premises, yields (has a consequence) another proposition Q, called the conclusion. Such an argument is denoted by

$$P1, P2, ..., Pn \vdash Q$$

The notion of a "logical argument" or "valid argument" is formalized as follows:

Definition 3.4: An argument P1, P2, ..., Pn $\vdash$ Q is said to be valid if Q is true whenever all the premises P1, P2,...,Pn are true.

An argument which is not valid is called fallacy.

EXAMPLE 3.4

(a) The following argument is valid:

$p, p \to q \vdash q$ **(Law of Detachment)**

The proof of this rule follows from the truth table in Fig. 3-5(a). Specifically, p and $p \to q$ are true simultaneously only in Case (row) 1, and in this case q is true.

(b) The following argument is a fallacy:

$$p \to q, q \vdash p$$

For $p \rightarrow q$ and q are both true in Case (row) 3 in the truth table in Fig. 3-5(a), but in this case p is false.

Now the propositions $P1, P2, \ldots, Pn$ are true simultaneously if and only if the proposition $P1 \wedge P2 \wedge \ldots Pn$ is true. Thus the argument $P1, P2, \ldots, Pn \vdash Q$ is valid if and only if Q is true whenever $P1 \wedge P2 \wedge \ldots \wedge Pn$ is true or, equivalently, if the proposition $(P1 \wedge P2 \wedge \ldots \wedge Pn) \rightarrow Q$ is a tautology. We state this result formally.

Theorem 3.2: The argument $P1, P2, \ldots, Pn \vdash Q$ is valid if and only if the proposition $(P1 \wedge P2 \ldots \wedge Pn) \rightarrow Q$ is a tautology.

We apply this theorem in the next example.

EXAMPLE 3.5 A fundamental principle of logical reasoning states:

"If p implies q and q implies r, then p implies r"

p	q	r	$p\rightarrow q$	$q\rightarrow r$	$[(p\rightarrow q) \wedge (q\rightarrow r)]$	$p\rightarrow r$	$[(p\rightarrow q) \wedge (q\rightarrow r)]\rightarrow(p\rightarrow r)$
T	T	T	T	T	T	T	T
T	T	F	T	F	F	F	T
T	F	T	F	T	F	T	T
T	F	F	F	T	F	F	T
F	T	T	T	T	T	T	T
F	T	F	T	F	F	T	T
F	F	T	T	T	T	T	T
F	F	F	T	T	T	T	T

Fig. 3-6

That is, the following argument is valid:

$p \rightarrow q, q \rightarrow r \vdash p \rightarrow r$ **(Law of Syllogism)**

This fact is verified by the truth table in Fig. 3-6 which shows that the following proposition is a tautology:

$$[(p \rightarrow q) \wedge (q \rightarrow r)] \rightarrow (p \rightarrow r)$$

EXAMPLE 3.6 Consider the following argument:

S1 : If a man is a bachelor, he is unhappy.

S2 : If a man is unhappy, he dies young.

S : Bachelors die young

Here the statement S below the line denotes the conclusion of the argument, and the statements S1 and S2 above the line denote the premises. We claim that the argument $S1, S2 \vdash S$ is valid. For the argument is of the form

$$p \rightarrow q, q \rightarrow r \vdash p \rightarrow r$$

where p is "He is a bachelor," q is "He is unhappy" and r is "He dies young;" and by Example 3.5 this argument (Law of Syllogism) is valid.

4.1 FUNCTIONS

4.1.1 Introduction

One of the most important concepts in mathematics is that of a function. The terms “map,” “mapping,” “transformation,” and many others mean the same thing; the choice of which word to use in a given situation is usually determined by tradition and the mathematical background of the person using the term.

4.1.2 Definition

Suppose that to each element of a set A we assign a unique element of a set B; the collection of such assignments is called a function from A into B. The set A is called the domain of the function, and the set B is called the target set or codomain.

Functions are ordinarily denoted by symbols. For example, let f denote a function from A into B. Then we write

$$f: A \to B$$

which is read: “f is a function from A into B,” or “f takes (or maps) A into B.” If $a \in A$, then f(a) (read: “f of a”) denotes the unique element of B which f assigns to a; it is called the image of a under f, or the value of f at a.

The set of all image values is called the range or image of f. The image of $f : A \to B$ is denoted by Ran(f), Im(f) or f (A).

Frequently, a function can be expressed by means of a mathematical formula. For example, consider the function which sends each real number into its square. We may describe this function by writing

$$f(x) = x^2 \ or \ y = x^2$$

In the first notation, x is called a variable and the letter f denotes the function. In the second notation, x is called the independent variable and y is called the dependent variable since the value of y will depend on the value of x.

EXAMPLE 4.1

(a) Consider the function $f (x) = x^3$, i.e., f assigns to each real number its cube. Then the image of 2 is 8, and so we may write f (2) = 8

(b) Figure 4-1 defines a function f from A = {a, b, c, d} into B = {r, s, t, u} in the obvious way. Here

$$f (a) = s, f (b) = u, f (c) = r, f (d) = s$$

The image of f is the set of image values, {r, s, u}. Note that t does not belong to the image of f because t is not the image of any element under f.

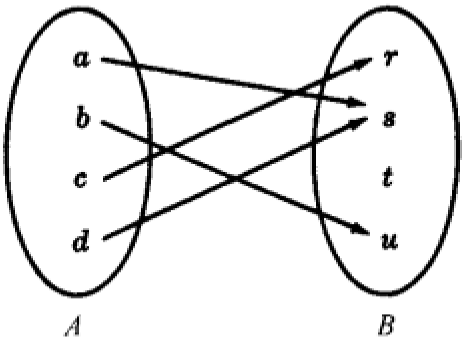


Fig. 4-1

4.2 ONE-TO-ONE, ONTO, AND INVERTIBLE FUNCTIONS

A function f : A → B is said to be **one-to-one** (written 1-1) if different elements in the domain A have distinct images. Another way of saying the same thing is that f is one-to-one if f(a) = f(a') implies a = a'.

A function f: A → B is said to be an **onto** function if each element of B is the image of some element of A. In other words, f : A → B is onto if the image of f is the entire codomain, i.e., if f (A) = B. In such a case we say that f is a function from A onto B or that f maps A onto B.

A function f: A → B is invertible if its inverse relation f^{-1} is a function from B to A. In general, the inverse relation f^{-1} may not be a function. The following theorem gives simple criteria which tells us when it is.

EXAMPLE 4.2

Consider the functions f_1: A → B, f_2: B → C, f_3: C → D and f_4: D → E defined by the diagram of Fig. 4-2. Now f_1 is one-to-one since no element of B is the image of more than one element of A. Similarly, f_2 is one-to-one. However, neither f_3 nor f_4 is one-to-one since $f_3(r) = f_3(u)$ and $f_4(v) = f_4(w)$.

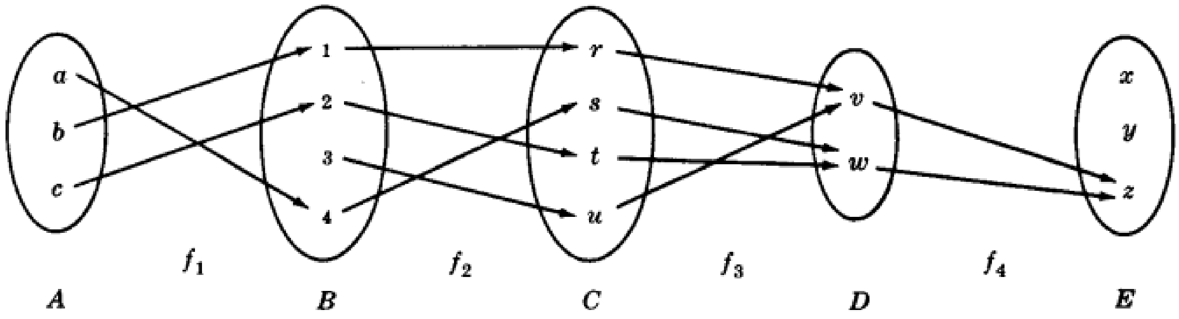


Fig. 4-2

As far as being onto is concerned, f_2 and f_3 are both onto functions since every element of C is the image under f_2 of some element of B and every element of D is the image under f_3 of some element of C, $f_2(B) = C$ and $f_3(C) = D$. On the other hand, f_1 is not onto since 3 ∈ B is not the image under f_1 of any element of A and f_4 is not onto since x ∈ E is not the image under f_4 of any element of D.

Thus, f_1 is one-to-one but not onto, f_3 is onto but not one-to-one and f_4 is neither one-to-one nor onto. However, f_2 is both one-to-one and onto, i.e., is a one-to-one correspondence between A and B. Hence f_2 is invertible and f_2^{-1} is a function from C to B.

5.1 PERMUTATIONS AND COMBINATIONS

This session develops some techniques for determining, without direct enumeration, the number of possible outcomes of a particular event or the number of elements in a set. Such sophisticated counting is sometimes called combinatorial analysis. It includes the study of permutations and combinations.

5.2 Basic Counting Principles

There are two basic counting principles used throughout this session. The first one involves addition and the second one multiplication.

Sum Rule Principle:

Suppose some event E can occur in m ways and a second event F can occur in n ways, and suppose both events cannot occur simultaneously. Then E or F can occur in m + n ways.

Product Rule Principle:

Suppose there is an event E which can occur in m ways and, independent of this event, there is a second event F which can occur in n ways. Then combinations of E and F can occur in mn ways.

The above principles can be extended to three or more events. That is, suppose an event E_1 can occur in n_1 ways, a second event E_2 can occur in n_2 ways, and, following E_2; a third event E_3 can occur in n_3 ways, and so on. Then:

5.2.1 Sum Rule: If no two events can occur at the same time, then one of the events can occur in:

$$n_1 + n_2 + n_3 + \cdots \text{ ways.}$$

5.2.2 Product Rule: If the events occur one after the other, then all the events can occur in the order indicated in:

$$n_1 . n_2 . n_3 . \ldots \text{ ways.}$$

EXAMPLE 5.1 Suppose a college has 3 different history courses, 4 different literature courses, and 2 different sociology courses.

(a) The number m of ways a student can choose one of each kind of courses is:

m = 3(4)(2) = 24

(b) The number n of ways a student can choose just one of the courses is:

n = 3 + 4 + 2 = 9

There is a set theoretical interpretation of the above two principles. Specifically, suppose n(A) denotes the number of elements in a set A. Then:

(1) **Sum Rule Principle:** Suppose A and B are disjoint sets. Then

$$n(A \cup B) = n(A) + n(B)$$

(2) **Product Rule Principle**: Let A × B be the Cartesian product of sets A and B. Then

$$n(A \times B) = n(A) \cdot n(B)$$

5.3 MATHEMATICAL FUNCTIONS

We discuss two important mathematical functions frequently used in combinatorics.

5.3.1 Factorial Function

The product of the positive integers from 1 to n inclusive is denoted by n!, read “n factorial.” Namely:

$$n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot (n-2)(n-1)n = n(n-1)(n-2) \cdot \ldots \cdot 3 \cdot 2 \cdot 1$$

Accordingly, 1! = 1 and n! = n(n − l)!. It is also convenient to define 0! = 1.

EXAMPLE 5.2

(a) 3! = 3 · 2 · 1 = 6, 4! = 4 · 3 · 2 · 1 = 24, 5 = 5 · 4! = 5(24) = 120.

(b) $\frac{12\,.\,11\,.\,10}{3\,.\,2\,.\,1} = \frac{12\,.\,11\,.\,10\,.\,9!}{3\,.\,2\,.\,1\,.\,9!} = \frac{12!}{3!9!}$ and, more generally,

$$\frac{n(n-1)\ldots(n-r+1)}{r(r-1)\ldots 3\,.\,2\,.\,1} = \frac{n(n-1)\ldots(n-r+1)(n-r)!}{r(r-1)\ldots 3\,.\,2\,.\,1\,.\,(n-r)!} = \frac{n!}{r!\,(n-r)!}$$

(c) For large n, one uses Stirling’s approximation (where e = 2.7128...):

$$n! = \sqrt{2\pi n}\, n^n e^{-n}$$

5.3.2 Binomial Coefficients

The symbol $\binom{n}{r}$, read “nCr” or “n Choose r,” where r and n are positive integers with r ≤ n, is defined as follows:

$$\binom{n}{r} = \frac{n(n-1)\ldots(n-r+1)}{r(r-1)\ldots 3\,.\,2\,.\,1} \quad \text{or equivalently} \quad \binom{n}{r} = \frac{n!}{r!\,(n-r)!}$$

Note that n − (n − r) = r. This yields the following important relation.

Lemma 5.1: $\binom{n}{n-r} = \binom{n}{r}$ or equivalently, $\binom{n}{a} = \binom{n}{b}$ where $a + b = n$.

Motivated by that fact that we defined 0! = 1, we define:

$$\binom{n}{0} = \frac{n!}{0!\,n!} = 1 \quad \text{and} \quad \binom{0}{0} = \frac{0!}{0!\,0!} = 1$$

EXAMPLE 5.3

(a) $\binom{8}{2} = \frac{8\,.\,7}{2\,.\,1} = 28$; $\binom{9}{4} = \frac{9\,.\,8\,.\,7\,.\,6}{4\,.\,3\,.\,2\,.\,1} = 126$; $\binom{12}{5} = \frac{12\,.\,11\,.\,10\,.\,9\,.\,8}{5\,.\,4\,.\,3\,.\,2\,.\,1} = 792.$

Note that $\binom{n}{r}$ has exactly r factors in both the numerator and the denominator.

5.3.3 Binomial Coefficients and Pascal's Triangle

The numbers $\binom{n}{r}$ are called binomial coefficients, since they appear as the coefficients in the expansion of $(a+b)^n$. Specifically:

Theorem (Binomial Theorem) 5.2: $(a+b)^n = \sum_{k=0}^{n} \binom{n}{r} a^{n-k} b^k$

The coefficients of the successive powers of a + b can be arranged in a triangular array of numbers, called Pascal's triangle, as pictured in Fig. 5-1. The numbers in Pascal's triangle have the following interesting properties:

(i) The first and last number in each row is 1.

(ii) Every other number can be obtained by adding the two numbers appearing above it. For example:

$$10 = 4 + 6,\ 15 = 5 + 10,\ 20 = 10 + 10.$$

Since these numbers are binomial coefficients, we state the above property formally.

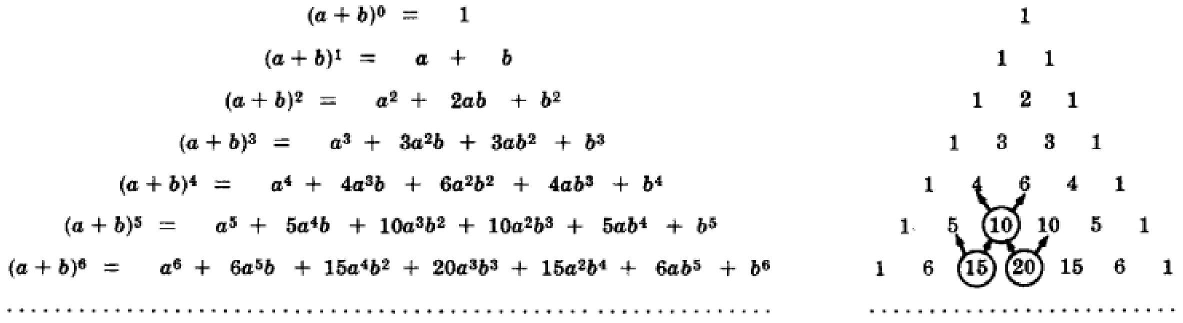

$(a+b)^0 = 1$
$(a+b)^1 = a + b$
$(a+b)^2 = a^2 + 2ab + b^2$
$(a+b)^3 = a^3 + 3a^2b + 3ab^2 + b^3$
$(a+b)^4 = a^4 + 4a^3b + 6a^2b^2 + 4ab^3 + b^4$
$(a+b)^5 = a^5 + 5a^4b + 10a^3b^2 + 10a^2b^3 + 5ab^4 + b^5$
$(a+b)^6 = a^6 + 6a^5b + 15a^4b^2 + 20a^3b^3 + 15a^2b^4 + 6ab^5 + b^6$
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1

Fig. 5-1 Pascal's triangle

5.4 PERMUTATIONS

Any arrangement of a set of n objects in a given order is called a permutation of the object (taken all at a time). Any arrangement of any r ≤ n of these objects in a given order is called an "r-permutation" or "a permutation of the n objects taken r at a time." Consider, for example, the set of letters A, B, C, D. Then:

(i) BDCA, DCBA, and ACDB are permutations of the four letters (taken all at a time).

(ii) BAD, ACB, DBC are permutations of the four letters taken three at a time.

(iii) AD, BC, CA are permutations of the four letters taken two at a time.

We usually are interested in the number of such permutations without listing them.

The number of permutations of n objects taken r at a time will be denoted by

P (n, r) (other texts may use $_nP_r$, $P_{n,r}$, or $(n)_r$).

The following theorem applies.

Theorem 5.1: $P(n, r) = n(n-1)(n-2)\ldots(n-r+1) = \frac{n!}{(n-r)!}$

We emphasize that there are r factors in n(n − 1)(n − 2)···(n − r + 1).

EXAMPLE 5.4 Find the number m of permutations of six objects, say, A, B, C, D, E, F, taken three at a time. In other words, find the number of "three-letter words" using only the given six letters without repetition.

Let us represent the general three-letter word by the following three positions:

$$\underline{\quad\quad}, \underline{\quad\quad}, \underline{\quad\quad}$$

The first letter can be chosen in 6 ways; following this the second letter can be chosen in 5 ways; and, finally, the third letter can be chosen in 4 ways. Write each number in its appropriate position as follows:

$$\underline{\;6\;}, \underline{\;5\;}, \underline{\;4\;}$$

By the Product Rule there are m = 6 · 5 · 4 = 120 possible three-letter words without repetition from the six letters. Namely, there are 120 permutations of 6 objects taken 3 at a time. This agrees with the formula in Theorem 5.1:

P (6, 3) = 6 · 5 · 4 = 120

In fact, Theorem 5.1 is proven in the same way as we did for this particular case.

Consider now the special case of P (n, r) when r = n. We get the following result.

Corollary 5.1: There are n! permutations of n objects (taken all at a time).

For example, there are 3! = 6 permutations of the three letters A, B, C. These are:

ABC, ACB, BAC, BCA, CAB, CBA.

5.4.1 Permutations with Repetitions

Frequently we want to know the number of permutations of a multiset, that is, a set of objects some of which are alike. We will let

P (n; n_1, n_2, ..., n_r)

denote the number of permutations of n objects of which n_1 are alike, n_2 are alike, ..., n_r are alike. The general formula follows:

Theorem 5.2: $P(n;\ n_1, n_2, \ldots, n_r) = \frac{n!}{n_1! n_2! \ldots n_r!}$

We indicate the proof of the above theorem by a particular example. Suppose we want to form all possible five-letter "words" using the letters from the word "BABBY." Now there are 5! = 120 permutations of the objects B_1, A, B_2, B_3, Y, where the three B's are distinguished. Observe that the following six permutations

$$B_1B_2B_3AY,\ B_2B_1B_3AY,\ B_3B_1B_2AY,\ B_1B_3B_2AY,\quad B_2B_3B_1AY,\ B_3B_2B_1AY,$$

produce the same word when the subscripts are removed. The 6 comes from the fact that there are 3! = 3·2·1 = 6 different ways of placing the three B's in the first three positions in the permutation. This is true for each set of three positions in which the B's can appear. Accordingly, the number of different five-letter words that can be formed using the letters from the word "BABBY" is:

$$P(5;3) = \frac{5!}{3!} = 20$$

EXAMPLE 5.5 Find the number m of seven-letter words that can be formed using the letters of the word "BENZENE."

We seek the number of permutations of 7 objects of which 3 are alike (the three E's), and 2 are alike (the two N's). By Theorem 5.2,

$$m = P(7;3,2) = \frac{7!}{3!\,2!} = \frac{7\,.\,6\,.\,5\,.\,4\,.\,3\,.\,2\,.\,1}{3\,.\,2\,.\,1\,.\,2\,.\,1} = 420$$

5.5 Ordered Samples

Many problems are concerned with choosing an element from a set S, say, with n elements. When we choose one element after another, say, r times, we call the choice an ordered sample of size r. We consider two cases.

(1) Sampling with replacement

Here the element is replaced in the set S before the next element is chosen. Thus, each time there are n ways to choose an element (repetitions are allowed). The Product rule tells us that the number of such samples is:

$$n.n.n\ldots n.n(r \text{ factors}) = n^r$$

(2) Sampling without replacement

Here the element is not replaced in the set S before the next element is chosen. Thus, there is no repetition in the ordered sample. Such a sample is simply an r-permutation. Thus the number of such samples is:

$$P(n.r) = n(n-1)(n-2)\ldots(n-r+1) = \frac{n!}{(n-r)!}$$

EXAMPLE 5.6 Three cards are chosen one after the other from a 52-card deck. Find the number m of ways this can be done: (a) with replacement; (b) without replacement.

(a) Each card can be chosen in 52 ways. Thus m = 52(52)(52) = 140 608.

(b) Here there is no replacement. Thus the first card can be chosen in 52 ways, the second in 51 ways, and the third in 50 ways. Therefore:

m = P (52, 3) = 52(51)(50) = 132 600

5.6 COMBINATIONS

Let S be a set with n elements. A combination of these n elements taken r at a time is any selection of r of the elements where order does not count. Such a selection is called an r-combination; it is simply a subset of S with r elements. The number of such combinations will be denoted by

C(n, r) (other texts may use $_nC_r$, $C_{n,r}$, or C_r^n).

Theorem 5.3: $C(n,r) = \frac{P(n,\ r)}{r!} = \frac{n!}{r!(n-r)!}$

Recall that the binomial coefficient $\binom{n}{r}$ was defined to be $\frac{n!}{r!(n-r)!}$; hence

$$\boxed{\mathbf{C(r,n)} = \binom{\mathbf{n}}{\mathbf{r}}}$$

We shall use $C(n,r)$ and $\binom{n}{r}$ interchangeably.

EXAMPLE 5.8 A farmer buys 3 cows, 2 pigs, and 4 hens from a man who has 6 cows, 5 pigs, and 8 hens. Find the number m of choices that the farmer has.

The farmer can choose the cows in C(6, 3) ways, the pigs in C(5, 2) ways, and the hens in C(8, 4) ways. Thus the number m of choices follows:

$$m = \binom{6}{3}\binom{5}{2}\binom{8}{4} = \frac{6\,.\,5\,.4}{3\,.\,2\,.\,1}.\frac{5\,.\,4}{2\,.\,1}.\frac{8\,.\,7\,.\,6\,.\,5}{4\,.\,3\,.\,2\,.\,1} = 20\,.\,10\,.\,70 = 14\ 000$$

5.7 THE PIGEONHOLE PRINCIPLE

Many results in combinational theory come from the following almost obvious statement.

5.7.1 Pigeonhole Principle: If n pigeonholes are occupied by n + 1 or more pigeons, then at least one pigeonhole is occupied by more than one pigeon.

This principle can be applied to many problems where we want to show that a given situation can occur.

EXAMPLE 5.9

(a) Suppose a department contains 13 professors, then two of the professors (pigeons) were born in the same month (pigeonholes).

(b) Find the minimum number of elements that one needs to take from the set S = {1, 2, 3,..., 9} to be sure that two of the numbers add up to 10.

Here the pigeonholes are the five sets {1, 9}, {2, 8}, {3, 7}, {4, 6}, {5}. Thus any choice of six elements (pigeons) of S will guarantee that two of the numbers add up to ten.

The Pigeonhole Principle is generalized as follows.

5.7.2 Generalized Pigeonhole Principle: If n pigeonholes are occupied by $kn + 1$ or more pigeons, where k is a positive integer, then at least one pigeonhole is occupied by $k + 1$ or more pigeons.

EXAMPLE 5.10 Find the minimum number of students in a class to be sure that three of them are born in the same month.

Here the $n = 12$ months are the pigeonholes, and $k + 1 = 3$ so $k = 2$. Hence among any $kn + 1 = 25$ students (pigeons), three of them are born in the same month.

5.8 THE INCLUSION–EXCLUSION PRINCIPLE

Let A and B be any finite sets. Given:

$$n(A \cup B) = n(A) + n(B) - n(A \cap B)$$

In other words, to find the number $n(A \cup B)$ of elements in the union of A and B, we add n(A) and n(B) and then we subtract $n(A \cap B)$; that is, we “include” n(A) and n(B), and we “exclude” $n(A \cap B)$. This follows from the fact that, when we add n(A) and n(B), we have counted the elements of $(A \cap B)$ twice.

The above principle holds for any number of sets. We first state it for three sets.

Theorem 5.4: For any finite sets A, B, C we have

$$n(A \cup B \cup C) = n(A) + n(B) + n(C) - n(A \cap B) - n(A \cap C) - n(B \cap C) + n(A \cap B \cap C)$$

That is, we “include” n(A), n(B), n(C), we “exclude” $n(A \cap B)$, $n(A \cap C)$, $n(B \cap C)$, and finally “include” $n(A \cap B \cap C)$.

EXAMPLE 5.11 Find the number of mathematics students at a college taking at least one of the languages French, German, and Russian, given the following data:

65 study French,	20 study French and German,	
45 study German,	25 study French and Russian,	8 study all three languages.
42 study Russian,	15 study German and Russian,	

We want to find $n(F \cup G \cup R)$ where F, G, and R denote the sets of students studying French, German, and Russian, respectively.

By the Inclusion–Exclusion Principle,

$$n(F \cup G \cup R) = n(F) + n(G) + n(R) - n(F \cap G) - n(F \cap R) - n(G \cap R) + n(F \cap G \cap R)$$

$$= 65 + 45 + 42 - 20 - 25 - 15 + 8 = 100$$

Namely, 100 students study at least one of the three languages.

6.1 GRAPH THEORY

6.2 GRAPHS AND MULTIGRAPHS

A graph G consists of two things:

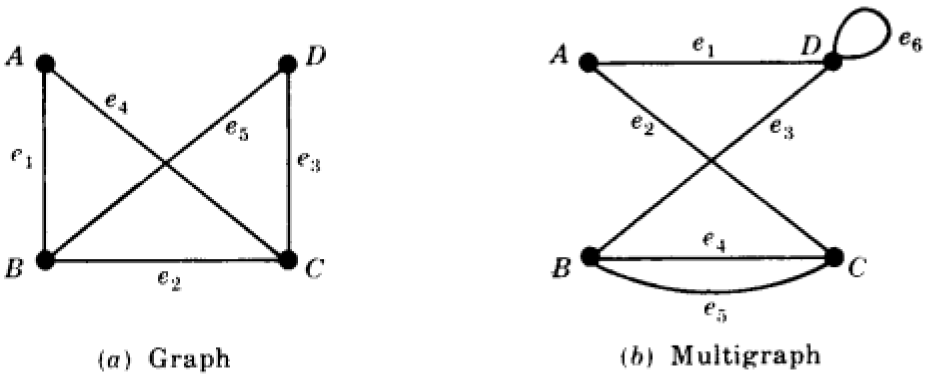
i. A set V = V (G) whose elements are called vertices, points, or nodes of G.
ii. A set E = E(G) of unordered pairs of distinct vertices called edges of G.

We denote such a graph by G(V , E) when we want to emphasize the two parts of G.

Vertices u and v are said to be adjacent or neighbors if there is an edge e = {u, v}. In such a case, u and v are called the endpoints of e, and e is said to connect u and v. Also, the edge e is said to be incident on each of its endpoints u and v. Graphs are pictured by diagrams in the plane in a natural way. Specifically, each vertex v in V is represented by a dot (or small circle), and each edge e = {v1, v2} is represented by a curve which connects its endpoints v1 and v2 For example, Fig. 6-1(a) represents the graph G(V , E) where:

(i) V consists of vertices A, B, C, D.
(ii) E consists of edges e1 = {A, B}, e2 = {B,C}, e3 = {C, D}, e4 = {A, C}, e5 = {B,D}.

In fact, we will usually denote a graph by drawing its diagram rather than explicitly listing its vertices and edges.



(a) Graph

(b) Multigraph

Fig. 6-1

6.2.1 Multigraphs

Consider the diagram in Fig. 6-1(b). The edges e4 and e5 are called multiple edges since they connect the same endpoints, and the edge e6 is called a loop since its endpoints are the same vertex. Such a diagram is called a multigraph; the formal definition of a graph permits neither multiple edges nor loops. Thus a graph may be defined to be a multigraph without multiple edges or loops.

Remark: Some texts use the term graph to include multigraphs and use the term simple graph to mean a graph without multiple edges and loops.

6.2.2 Degree of a Vertex

The degree of a vertex v in a graph G, written deg (v), is equal to the number of edges in G which contain v, that is, which are incident on v. Since each edge is counted twice in counting the degrees of the vertices of G, we have the following simple but important result.

Theorem 6.1: The sum of the degrees of the vertices of a graph G is equal to twice the number of edges in G. Consider, for example, the graph in Fig. 6-1(a). We have

deg(A) = 2, deg(B) = 3, deg(C) = 3, deg(D) = 2.

The sum of the degrees equals 10 which, as expected, is twice the number of edges. A vertex is said to be even or odd according as its degree is an even or an odd number. Thus A and D are even vertices whereas B and C are odd vertices.

Theorem 6.1 also holds for multigraphs where a loop is counted twice toward the degree of its endpoint. For example, in Fig. 6-1(b) we have deg(D) = 4 since the edge e6 is counted twice; hence D is an even vertex. A vertex of degree zero is called an isolated vertex.

6.2.3 Finite Graphs, Trivial Graph

A multigraph is said to be finite if it has a finite number of vertices and a finite number of edges. Observe that a graph with a finite number of vertices must automatically have a finite number of edges and so must be finite. The finite graph with one vertex and no edges, i.e., a single point, is called the trivial graph. Unless otherwise specified, the multigraphs in this course shall be finite.

6.3 SUBGRAPHS, ISOMORPHIC AND HOMEOMORPHIC GRAPHS

This section will discuss important relationships between graphs.

6.3.1 Subgraphs

Consider a graph G = G(V , E). A graph H = H (V', E') is called a subgraph of G if the vertices and edges of H are contained in the vertices and edges of G, that is, if $V' \subseteq$ V and $E' \subseteq$ E. In particular:

(i) A subgraph H (V', E') of G(V , E) is called the subgraph induced by its vertices V if its edge set E contains all edges in G whose endpoints belong to vertices in H.

(ii) If v is a vertex in G, then G − v is the subgraph of G obtained by deleting v from G and deleting all edges in G which contain v.

(iii) If e is an edge in G, then G − e is the subgraph of G obtained by simply deleting the edge e from G.

6.3.2 Isomorphic Graphs

Graphs G(V, E) and G(V^*, E^*) are said to be isomorphic if there exists a one-to-one correspondence f : V → V^* such that {u, v} is an edge of G if and only if {f (u), f (v)} is an edge of G^*. Normally, we do not distinguish between isomorphic graphs (even though their diagrams may "look different"). Figure 6-2 gives ten graphs pictured as letters. We note that A and R are isomorphic graphs. Also, F and T are isomorphic graphs, K and X are isomorphic graphs and M, S, V , and Z are isomorphic graphs.

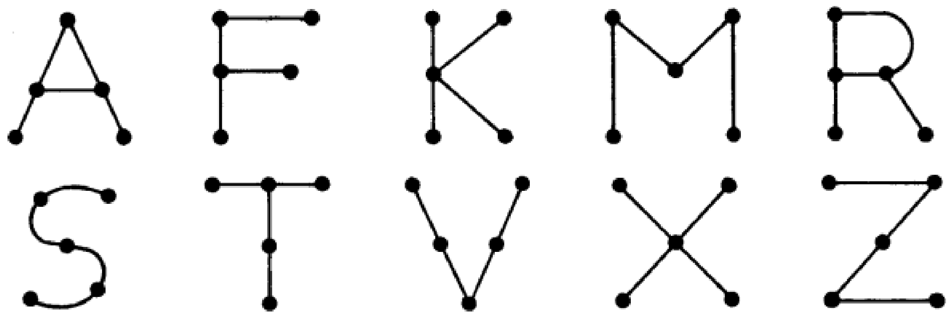


Fig. 6-2

6.3.4 Homeomorphic Graphs

Given any graph G, we can obtain a new graph by dividing an edge of G with additional vertices. Two graphs G and G* are said to homeomorphic if they can be obtained from the same graph or isomorphic graphs by this method. The graphs (a) and (b) in Fig. 6-3 are not isomorphic, but they are homeomorphic since they can be obtained from the graph (c) by adding appropriate vertices.

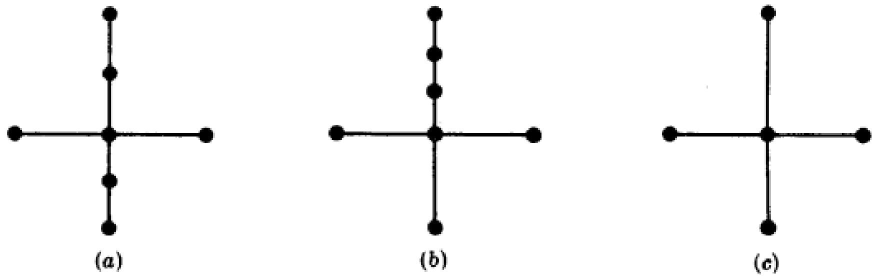


Fig. 6-3

6.4 PATHS, CONNECTIVITY

A path in a multigraph G consists of an alternating sequence of vertices and edges of the form

$$v_0, e_1, v_1, e_2, v_2, ..., e_{n-1}, v_{n-1}, e_n, v_n$$

where each edge e_i contains the vertices v_{i-1} and v_i (which appear on the sides of e_i in the sequence). The number n of edges is called the length of the path. When there is no ambiguity, we denote a path by its sequence of vertices (v_0, v_1,..., v_n). The path is said to be closed if v_0 = v_n. Otherwise, we say the path is from v_0, to v_n or between v_0 and v_n, or connects v_0 to v_n.

A **simple path** is a path in which all vertices are distinct. (A path in which all edges are distinct will be called a **trail**.) A cycle is a closed path of length 3 or more in which all vertices are distinct except v_0 = v_n. A cycle of length k is called a k-cycle.

EXAMPLE 6.1 Consider the graph G in Fig. 6-4(a). Consider the following sequences:

α = (P4, P1, P2, P5, P1, P2, P3, P6), β = (P4, P1, P5, P2, P6),

γ = (P4, P1, P5, P2, P3, P5, P6), δ = (P4, P1, P5, P3, P6).

The sequence α is a path from P4 to P6; but it is not a trail since the edge {P1, P2} is used twice. The sequence β is not a path since there is no edge {P2, P6}. The sequence γ is a trail since no edge is used twice; but it is not a simple path since the vertex P5 is used twice. The sequence δ is a simple path from P4 to P6; but it is not the shortest path (with respect to length) from P4 to P6. The shortest path from P4 to P6 is the simple path (P4, P5, P6) which has length 2.

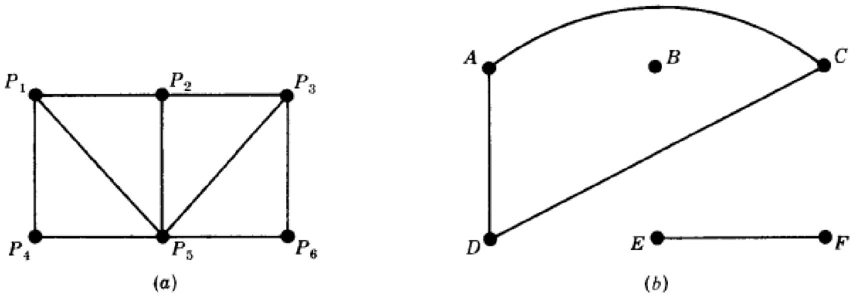


Fig. 6-4

By eliminating unnecessary edges, it is not difficult to see that any path from a vertex u to a vertex v can be replaced by a simple path from u to v. We state this result formally.

Theorem 6.2: There is a path from a vertex u to a vertex v if and only if there exists a simple path from u to v.

6.4.1 Connectivity, Connected Components

A graph G is connected if there is a path between any two of its vertices. The graph in Fig. 4-4(a) is connected, but the graph in Fig. 6-4(b) is not connected since, for example, there is no path between vertices D and E.

Suppose G is a graph. A connected subgraph H of G is called a connected component of G if H is not contained in any larger connected subgraph of G. It is intuitively clear that any graph G can be partitioned into its connected components. For example, the graph G in Fig. 6-4(b) has three connected components, the subgraphs induced by the vertex sets {A, C, D}, {E,F}, and {B}.

The vertex B in Fig. 6-4(b) is called an isolated vertex since B does not belong to any edge or, in other words, deg(B) = 0. Therefore, as noted, B itself forms a connected component of the graph.

Remark: Formally speaking, assuming any vertex u is connected to itself, the relation “u is connected to v” is an equivalence relation on the vertex set of a graph G and the equivalence classes of the relation form the connected components of G.

6.4.2 Distance and Diameter

Consider a connected graph G. The distance between vertices u and v in G, written d(u, v), is the length of the shortest path between u and v. The diameter of G, written diam(G), is the maximum distance between any two points in G. For example, in Fig. 6-5(a), d(A, F) = 2 and diam(G) = 3, whereas in Fig. 6-5(b), d(A, F) = 3 and diam(G) = 4.

6.4.3 Cutpoints and Bridges

Let G be a connected graph. A vertex v in G is called a cutpoint if G − v is disconnected. (Recall that G − v is the graph obtained from G by deleting v and all edges containing v.) An edge e of G is called a bridge if G − e is disconnected. (Recall that G − e is the graph obtained from G by simply deleting the edge e). In Fig. 6-5(a), the vertex D is a cutpoint and there are no bridges. In Fig. 6-5 (b), the edge = {D, F} is a bridge. (Its endpoints D and F are necessarily cutpoints.)

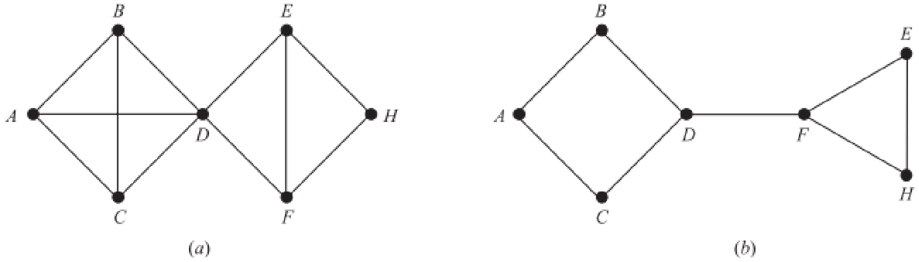


Fig. 6-5

6.5 LABELED AND WEIGHTED GRAPHS

A graph G is called a labeled graph if its edges and/or vertices are assigned data of one kind or another. In particular, G is called a weighted graph if each edge e of G is assigned a nonnegative number w(e) called the weight or length of v. Figure 6-6 shows a weighted graph where the weight of each edge is given in the obvious way. The weight (or length) of a path in such a weighted graph G is defined to be the sum of the weights of the edges in the path. One important problem in graph theory is to find a shortest path, that is, a path of minimum weight (length), between any two given vertices. The length of a shortest path between P and Q in Fig. 6-6 is 14; one such path is

(P , A1, A2, A5, A3, A6, Q)

The reader can try to find another shortest path.

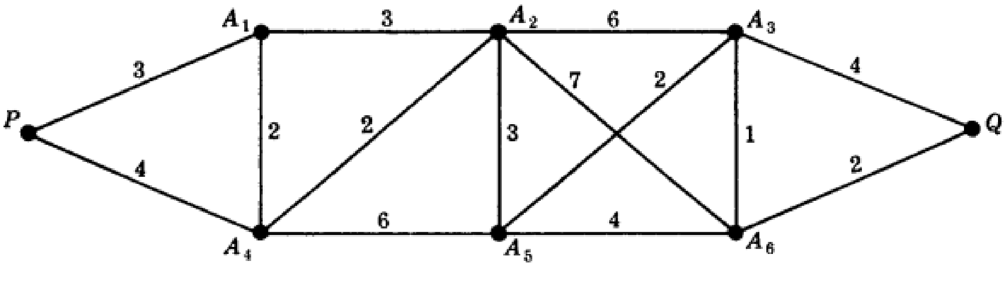


Fig. 6-6

6.6 COMPLETE, REGULAR, AND BIPARTITE GRAPHS

There are many different types of graphs. This section considers three of them: complete, regular, and bipartite graphs.

6.6.1 Complete Graphs

A graph G is said to be complete if every vertex in G is connected to every other vertex in G. Thus a complete graph G must be connected. The complete graph with n vertices is denoted by Kn. Figure 4-7 shows the graphs K1 through K6.

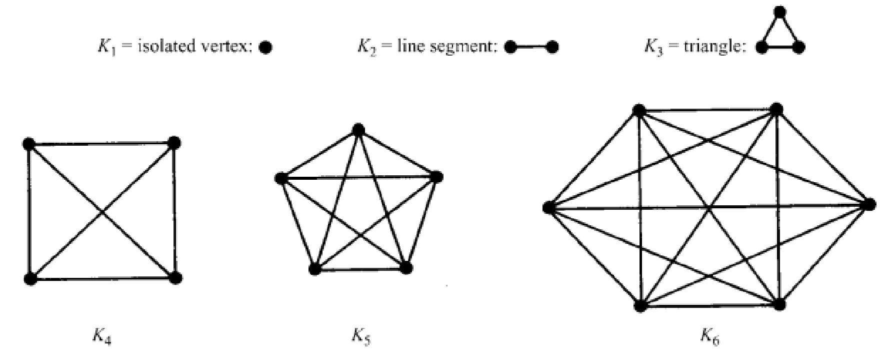


Fig. 6-7

6.6.2 Regular Graphs

A graph G is regular of degree k or k-regular if every vertex has degree k. In other words, a graph is regular if every vertex has the same degree. The connected regular graphs of degrees 0, 1, or 2 are easily described.

The connected 0-regular graph is the trivial graph with one vertex and no edges. The connected 1-regular graph is the graph with two vertices and one edge connecting them. The connected 2-regular graph with n vertices is the graph which consists of a single n-cycle. See Fig. 6-8.

The 3-regular graphs must have an even number of vertices since the sum of the degrees of the vertices is an even number (Theorem 6.1). Figure 6-9 shows two connected 3-regular graphs with six vertices. In general, regular graphs can be quite complicated. For example, there are nineteen 3-regular graphs with ten vertices. We note that the complete graph with n vertices K_n is regular of degree n − 1.

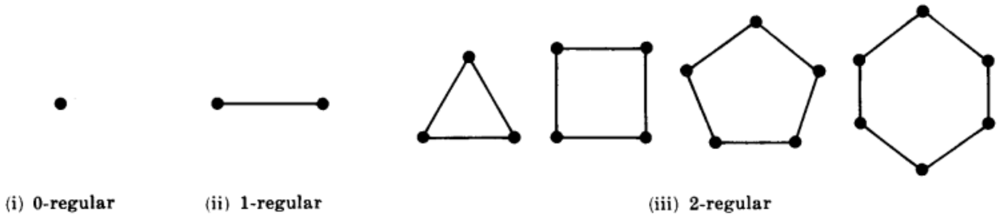


Fig. 6-8

6.6.3 Bipartite Graphs

A graph G is said to be bipartite if its vertices V can be partitioned into two subsets M and N such that each edge of G connects a vertex of M to a vertex of N. By a complete bipartite graph, we mean that each vertex of M is connected to each vertex of N; this graph is denoted by K_{mn} where m is the number of vertices in M and n is the number of vertices in N, and, for standardization, we will assume m ≤ n. Figure 6-10 shows the graphs $K_{2,3}$, $K_{3,3}$, and $K_{2,4}$, Clearly the graph Km,n has mn edges.

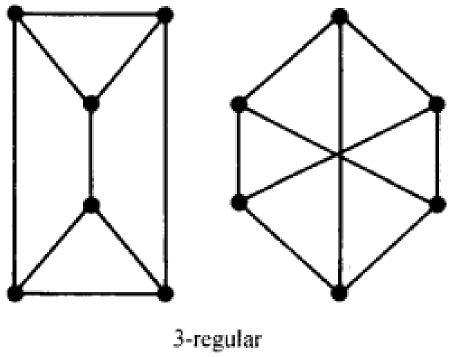


Fig. 6-9

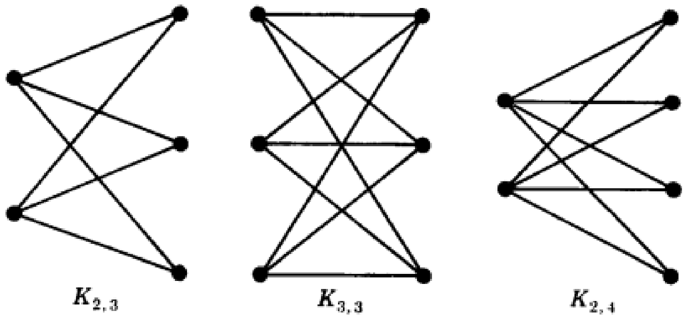


Fig. 6-10

6.7 TREE GRAPHS

A graph T is called a tree if T is connected and T has no cycles. Examples of trees are shown in Fig. 6-11. A forest G is a graph with no cycles; hence the connected components of a forest G are trees. A graph without cycles is said to be cycle-free. The tree consisting of a single vertex with no edges is called the degenerate tree.

Consider a tree T. Clearly, there is only one simple path between two vertices of T ; otherwise, the two paths would form a cycle. Also:

a. Suppose there is no edge {u, v} in T and we add the edge e = {u, v} to T . Then the simple path from u to v in T and e will form a cycle; hence T is no longer a tree.
b. On the other hand, suppose there is an edge e = {u, v} in T , and we delete e from T . Then T is no longer connected (since there cannot be a path from u to v); hence T is no longer a tree.

The following theorem applies when our graphs are finite.

Theorem 4.3: Let G be a graph with n > 1 vertices. Then the following are equivalent:

(i) G is a tree.
(ii) G is a cycle-free and has n − 1 edges.
(iii)G is connected and has n − 1 edges.

This theorem also tells us that a finite tree T with n vertices must have n − 1 edges. For example, the tree in Fig. Fig. 6-11(a) has 9 vertices and 8 edges, and the tree in F Fig. 6-11(b) has 13 vertices and 12 edges.

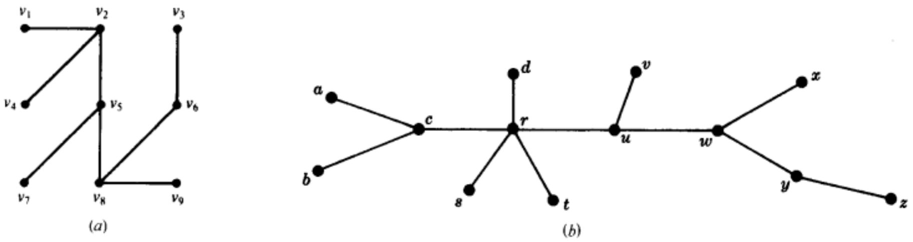


Fig. 6-11

6.7.1 Spanning Trees

A subgraph T of a connected graph G is called a spanning tree of G if T is a tree and T includes all the vertices of G. Figure 6-12 shows a connected graph G and spanning trees T1, T2, and T3 of G.

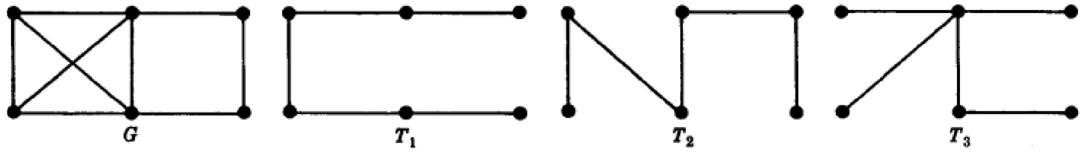


Fig. 6-12

6.7.2 Minimum Spanning Trees

Suppose G is a connected weighted graph. That is, each edge of G is assigned a nonnegative number called the weight of the edge. Then any spanning tree T of G is assigned a total weight obtained by adding the weights of the edges in T. A minimal spanning tree of G is a spanning tree whose total weight is as small as possible.

Algorithms 6.1 and 6.2 enable us to find a minimal spanning tree T of a connected weighted graph G where G has n vertices. (In which case T must have n − 1 vertices.)

Algorithm 6.1: The input is a connected weighted graph G with n vertices.

Step 1. Arrange the edges of G in the order of decreasing weights.

Step 2. Proceeding sequentially, delete each edge that does not disconnect the graph until n-1 edges remain.

Step 3. Exit.

Algorithm 6.2: The input is a connected weighted graph G with n vertices.

Step 1. Arrange the edges of G in the order of increasing weights.

Step 2. Starting only with the vertices of G and proceeding sequentially, add each edge which does not result in a cycle until n-1 edges are added.

Step 3. Exit.

The weight of a minimal spanning tree is unique, but the minimal spanning tree itself is not. Different minimal spanning trees can occur when two or more edges have the same weight. In such a case, the arrangement of the edges in Step 1 of Algorithms 6.1 or 6.2 is not unique and hence may result in different minimal spanning trees.

EXAMPLE 4.2 Find a minimal spanning tree of the weighted graph Q in Fig. 6-14(a). Note that Q has six vertices, so a minimal spanning tree will have five edges.

(a) Here we apply Algorithm 6.1. First we order the edges by decreasing weights, and then we successively delete edges without disconnecting Q until five edges remain. This yields the following data:

Edges	BC	AF	AC	BE	CE	BF	AE	DF	BD
Weight	8	7	7	7	6	5	4	4	3
Delete	Yes	Yes	Yes	No	No	Yes			

Thus the minimal spanning tree of Q which is obtained contains the edges BE, CE, AE, DF, BD The spanning tree has weight 24 and it is shown in Fig. 6-14(b).

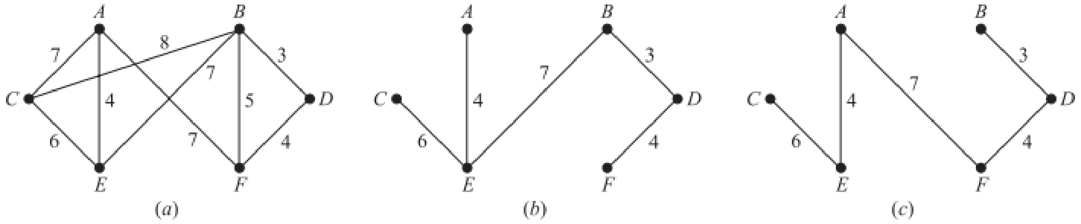


Fig. 6-14

(b) Here we apply Algorithm 4.2.

First we order the edges by increasing weights, and then we successively add edges without forming any cycles until five edges are included. This yields the following data:

Edges	BD	AE	DF	BF	CE	AC	AF	BE	BC
Weight	3	4	4	5	6	7	7	7	8
Add?	Yes	Yes	Yes	No	Yes	No	Yes		

Thus the minimal spanning tree of Q which is obtained contains the edges

BD, AE, DF , CE, AF

The spanning tree appears in Fig. 6-14(c). Observe that this spanning tree is not the same as the one obtained using Algorithm 6.1 as expected it also has weight 24.

6.8 PLANAR GRAPHS

A graph or multigraph which can be drawn in the plane so that its edges do not cross is said to be planar. Although the complete graph with four vertices K_4 is usually pictured with crossing edges as in Fig. 6-15(a), it can also be drawn with noncrossing edges as in Fig. 6-15(b); hence

K_4 is planar. Tree graphs form an important class of planar graphs. This section introduces our reader to these important graphs.

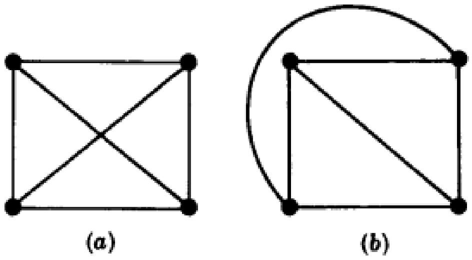


Fig. 6-15

6.9 REPRESENTING GRAPHS IN COMPUTER MEMORY

There are two standard ways of maintaining a graph G in the memory of a computer. One way, called the sequential representation of G, is by means of its adjacency matrix A. The other way, called the linked representation or adjacency structure of G, uses linked lists of neighbors. Matrices are usually used when the graph G is dense, and linked lists are usually used when G is sparse. (A graph G with m vertices and n edges is said to be dense when m = $O(n^2)$ and sparse when m = O(n) or even O(n log n)).

Regardless of the way one maintains a graph G in memory, the graph G is normally input into the computer by its formal definition, that is, as a collection of vertices and a collection of pairs of vertices (edges).

6.9.1 Adjacency Matrix

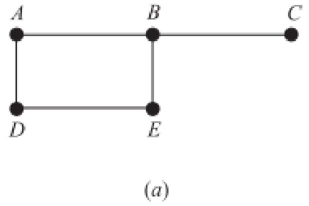
Suppose G is a graph with m vertices, and suppose the vertices have been ordered, say, $v_1, v_2, \ldots, v_m$. Then the adjacency matrix A = $[a_{ij}]$ of the graph G is the m × m matrix defined by

$$a_{ij} = \begin{cases} 1 & \text{if } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases}$$

Figure 6-16(b) contains the adjacency matrix of the graph G in Fig. 6-16(a) where the vertices are ordered A, B, C, D, E. Observe that each edge $\{v_i, v_j\}$ of G is represented twice, by a_{ij} = 1 and a_{ji} = 1. Thus, in particular, the adjacency matrix is symmetric.

The adjacency matrix A of a graph G does depend on the ordering of the vertices of G, that is, a different ordering of the vertices yields a different adjacency matrix. However, any two such adjacency matrices are closely related in that one can be obtained from the other by simply interchanging rows and columns. On the other hand, the adjacency matrix does not depend on the order in which the edges (pairs of vertics) are input into the computer.

There are variations of the above representation. If G is a multigraph, then we usually let a_{ij} denote the number of edges $\{v_i, v_j\}$. Moreover, if G is a weighted graph, then we may let a_{ij} denote the weight of the edge $\{v_i, v_j\}$.



$$\begin{array}{c|ccccc} & A & B & C & D & E \\ \hline A & 0 & 1 & 0 & 1 & 0 \\ B & 1 & 0 & 1 & 0 & 1 \\ C & 0 & 1 & 0 & 0 & 0 \\ D & 1 & 0 & 0 & 0 & 1 \\ E & 0 & 1 & 0 & 1 & 0 \end{array}$$

(b)

Fig. 6-16

6.9.2 Linked Representation of a Graph G

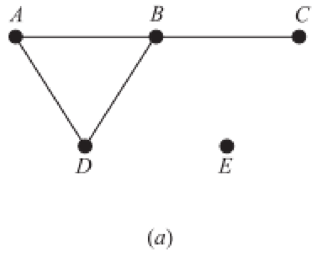
Let G be a graph with m vertices. The representation of G in memory by its adjacency matrix A has a number of major drawbacks. First of all it may be difficult to insert or delete vertices in G. The reason is that the size of A may need to be changed and the vertices may need to be reordered, so there may be many, many changes in the matrix A. Furthermore, suppose the number of edges is O(m) or even O(m log m), that is, suppose G is sparse. Then the matrix A will contain many zeros; hence a great deal of memory space will be wasted. Accordingly, when G is sparse, G is usually represented in memory by some type of linked representation, also called an adjacency structure, which is described below by means of an example.

Consider the graph G in Fig. 6-17(a). Observe that G may be equivalently defined by the table in Fig. 6-17(b) which shows each vertex in G followed by its adjacency list, i.e., its list of adjacent vertices (neighbors). Here the symbol ∅ denotes an empty list. This table may also be presented in the compact form

G = [A:B,D; B:A, C, D; C:B; D:A, B; E: ∅]

where a colon ":" separates a vertex from its list of neighbors, and a semicolon ";" separates the different lists.

Remark: Observe that each edge of a graph G is represented twice in an adjacency structure; that is, any edge, say {A, B}, is represented by B in the adjacency list of A, and also by A in the adjacency list of B. The graph G in Fig. 6-17(a) has four edges, and so there must be 8 vertices in the adjacency lists. On the other hand, each vertex in an adjacency list corresponds to a unique edge in the graph G.



Vertex	Adjacency list
A	*B, D*
B	*A, C, D*
C	*B*
D	*A, B*
E	∅

(b)

Fig. 6-17

The linked representation of a graph G, which maintains G in memory by using its adjacency lists, will normally contain two files (or sets of records), one called the Vertex File and the other called the Edge File, as follows.

(a) Vertex File: The Vertex File will contain the list of vertices of the graph G usually maintained by an array or by a linked list. Each record of the Vertex File will have the form

VERTEX	NEXT-V	PTR	▨

Here:

(1) VERTEX will be the name of the vertex.
(2) NEXT-V points to the next vertex in the list of vertices in the Vertex File when the vertices are maintained by a linked list, and
(3) PTR will point to the first element in the adjacency list of the vertex appearing in the Edge File.

The shaded area indicates that there may be other information in the record corresponding to the vertex.

(b) Edge File: The Edge File contains the edges of the graph G. Specifically; the Edge File will contain all the adjacency lists of G where each list is maintained in memory by a linked list. Each record of the Edge File will correspond to a vertex in an adjacency list and hence, indirectly, to an edge of G. The record will usually have the form

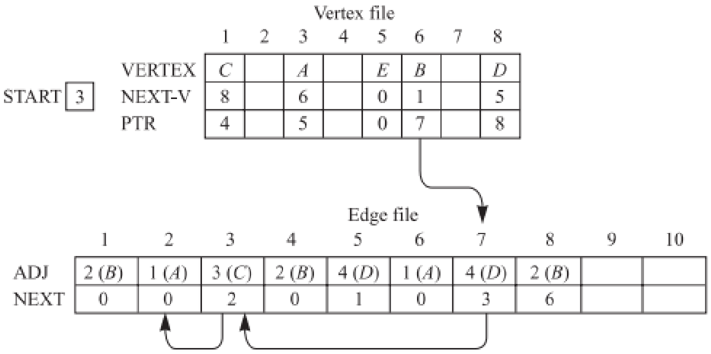
EDGE	ADJ	NEXT	▨

Here:

(1) EDGE will be the name of the edge (if it has one).
(2) ADJ points to the location of the vertex in the Vertex File.
(3) NEXT points to the location of the next vertex in the adjacency list.

We emphasize that each edge is represented twice in the Edge File, but each record of the file corresponds to a unique edge. The shaded area indicates that there may be other information in the record corresponding to the edge.

Figure 4-18 shows how the graph G in Fig. 6-17(a) may appear in memory. Here the vertices of G are maintained in memory by a linked list using the variable START to point to the first vertex. (Alternatively, one could use a linear array for the list of vertices, and then NEXT-V would not be required.) Note that the field EDGE is not needed here since the edges have no name. Figure 6-18 also shows, with the arrows, the adjacency list [D,C, A] of the vertex B.



Vertex file

START 3

	1	2	3	4	5	6	7	8
VERTEX	*C*		*A*		*E*	*B*		*D*
NEXT-V	8		6		0	1		5
PTR	4		5		0	7		8

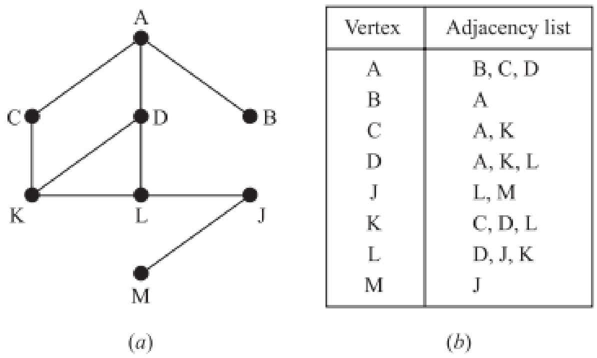
Edge file

	1	2	3	4	5	6	7	8	9	10
ADJ	2 (*B*)	1 (*A*)	3 (*C*)	2 (*B*)	4 (*D*)	1 (*A*)	4 (*D*)	2 (*B*)		
NEXT	0	0	2	0	1	0	3	6		

Fig. 6-18

6.10 GRAPH ALGORITHMS

This section discusses two important graph algorithms which systematically examine the vertices and edges of a graph G. One is called a depth-first search (DFS) and the other is called a breadth-first search (BFS). Any particular graph algorithm may depend on the way G is maintained in memory. Here we assume G is maintained in memory by its adjacency structure. Our test graph G with its adjacency structure appears in Fig. 6-19 where we assume the vertices are ordered alphabetically.



Vertex	Adjacency list
A	B, C, D
B	A
C	A, K
D	A, K, L
J	L, M
K	C, D, L
L	D, J, K
M	J

(*a*) (*b*)

Fig. 6-19

During the execution of our algorithms, each vertex (node) N of G will be in one of three states, called the status of N, as follows:

STATUS = 1: (Ready state) The initial state of the vertex N.

STATUS = 2: (Waiting state) The vertex N is on a (waiting) list, waiting to be processed.

STATUS = 3: (Processed state) The vertex N has been processed.

The waiting list for the depth-first search (DFS) will be a (modified) STACK (which we write horizontally with the top of STACK on the left), whereas the waiting list for the breadth-first search (BFS) will be a QUEUE.

6.10.1 Depth-first Search

The general idea behind a depth-first search beginning at a starting vertex A is as follows. First we process the starting vertex A. Then we process each vertex N along a path P which begins at A; that is, we process a neighbor of A, then a neighbor of A, and so on. After coming to a "dead end," that is to a vertex with no unprocessed neighbor, we backtrack on the path P until we can continue along another path P. And so on. The backtracking is accomplished by using a STACK to hold the initial vertices of future possible paths. We also need a field STATUS which tells us the current status of any vertex so that no vertex is processed more than once.

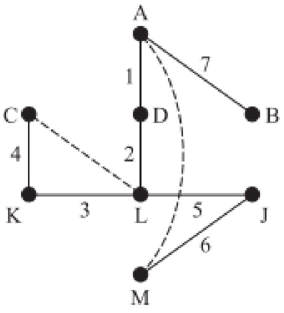
EXAMPLE 6.3 Suppose the DFS Algorithm is applied to the graph in Fig. 6-19. The vertices will be processed in the following order:

A, D, L, K, C, J, M, B

Specifically, Fig. 6-20(a) shows the sequence of vertices being processed and the sequence of waiting lists in STACK. (Note that after vertex A is processed, its neighbors, B, C, and D are added to STACK in the order first B, then C, and finally D; hence D is on the top of the STACK and D is the next vertex to be processed.) Each vertex, excluding A, comes from an adjacency list and hence corresponds to an edge of the graph. These edges form a spanning tree of G which is pictured in Fig. 6-20(b). The numbers indicate the order that the edges are added to the spanning tree, and the dashed lines indicate backtracking.

STACK	Vertex
A	A
D, C, B	D
L, K, C, B	L
K, J, ~~K~~, C, B	K
C, J, ~~C~~, B	C
J, B	J
M, B	M
B	B
∅	

(a)



(b)

Fig. 6-20

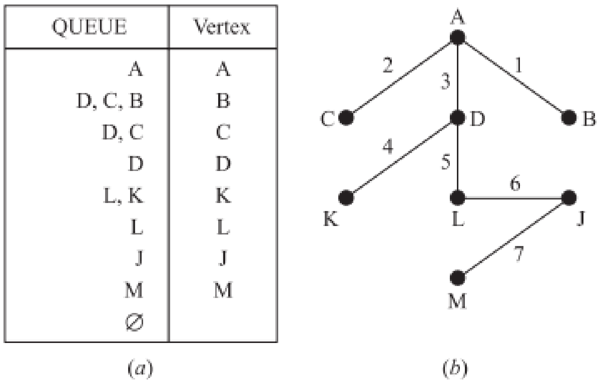
6.10.2 Breadth-first Search

The general idea behind a breadth-first search beginning at a starting vertex A is as follows. First we process the starting vertex A. Then we process all the neighbors of A. Then we process all the neighbors of neighbors of A. And so on. Naturally we need to keep track of the neighbors of a vertex, and we need to guarantee that no vertex is processed twice. This is accomplished by using a QUEUE to hold vertices that are waiting to be processed, and by a field STATUS which tells us the current status of a vertex.

EXAMPLE 4.4 Suppose the breadth-first search (BFS) Algorithm is applied to the graph in Fig. 6-19. The vertices will be processed in the following order:

A, B, C, D, K, L, J, M

Specifically, Fig. 6-21(a) shows the sequence of waiting lists in QUEUE and the sequence of vertices being processed (Note that after vertex A is processed, its neighbors, B, C, and D are added to QUEUE in the order first B, then C, and finally D; hence B is on the front of the QUEUE and so B is the next vertex to be processed.)Again, each vertex, excluding A, comes from an adjacency list and hence corresponds to an edge of the graph. These edges form a spanning tree of G which is pictured in Fig. 6-21(b). Again, the numbers indicate the order that the edges are added to the spanning tree. Observe that this spanning tree is different from the one in Fig. 6.20(b) which came from a depth-first search.



QUEUE	Vertex
A	A
D, C, B	B
D, C	C
D	D
L, K	K
L	L
J	J
M	M
∅	

(a)

(b)

Fig. 6-21

6.11 DIRECTED GRAPHS

A directed graph G or digraph (or simply graph) consists of two things:

(i) A set V whose elements are called vertices, nodes, or points.

(ii) A set E of ordered pairs (u, v) of vertices called arcs or directed edges or simply edges.

We will write G(V, E) when we want to emphasize the two parts of G. We will also write V (G) and E(G) to denote, respectively, the set of vertices and the set of edges of a graph G. (If it is not explicitly stated, the context usually determines whether or not a graph G is a directed graph.)

Suppose e = (u, v) is a directed edge in a digraph G. Then the following terminology is used:

(a) e begins at u and ends at v.

(b) u is the origin or initial point of e, and v is the destination or terminal point of e.

(c) v is a successor of u.

(d) u is adjacent to v, and v is adjacent from u.

If u = v, then e is called a loop.

The set of all successors of a vertex u is important; it is denoted and formally defined by

$$\text{succ}(u) = \{v \in V \mid \text{there exists an edge } (u, v) \in E\}$$

It is called the successor list or adjacency list of u.

If the edges and/or vertices of a directed graph G are labeled with some type of data, then G is called a labeled directed graph.

A directed graph (V, E) is said to be finite if its set V of vertices and its set E of edges are finite.

EXAMPLE 7.1

(a) Consider the directed graph G pictured in Fig. 7-1(a). It consists of four vertices, A, B, C, D, that is,

V (G) = {A, B, C, D} and the seven following edges:

E(G) = {e1, e2,...,e7}={(A, D), (B, A), (B, A), (D, B), (B, C), (D, C), (B, B)}

The edges e2 and e3 are said to be parallel since they both begin at B and end at A. The edge e7 is a loop since it begins and ends at B.

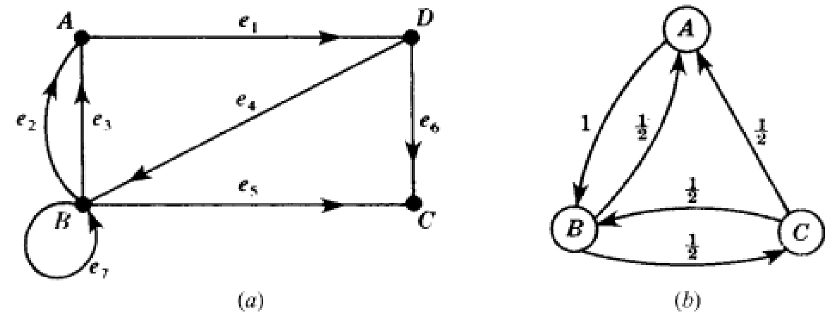


Fig. 7-1

(b) Suppose three boys, A, B, C, are throwing a ball to each other such that A always throws the ball to B, but B and C are just as likely to throw the ball to A as they are to each other. This dynamic system is pictured in Fig. 7-1(b) where edges are labeled with the respective probabilities, that is, A throws the ball to B with probability 1, B throws the ball to A and C each with probability 1/2, and C throws the ball to A and B each with probability 1/2.

6.11.1 Subgraphs

Let G = G(V , E) be a directed graph, and let V′ be a subset of the set V of vertices of G. Suppose E′ is a subset of E such that the endpoints of the edges in E′ belong to V′. Then H (V′, E′) is a directed graph, and it is called a subgraph of G. In particular, if E′ contains all the edges in E whose endpoints belong to V′, then H (V′, E′) is called the subgraph of G generated or determined by V′. For example, for the graph G = G(V , E) in Fig. 7-1(a), H (V′, E′) is the subgraph of G determine by the vertex set V′ where

V′ = {B, C, D} and E′ = (e4, e5, e6, e7) = {(D, B),(B, C),(D, C),(B, B)}

6.12 BASIC DEFINITIONS

This section discusses the degrees of vertices, paths, and connectivity in a directed graph.

6.12.1 Degrees

Suppose G is a directed graph. The outdegree of a vertex v of G, written outdeg(v), is the number of edges beginning at v, and the indegree of v, written indeg(v), is the number of edges ending at v. Since each edge begins and ends at a vertex we immediately obtain the following theorem.

Theorem 7.1: The sum of the outdegrees of the vertices of a digraph G equals the sum of the indegrees of the vertices, which equals the number of edges in G.

A vertex v with zero indegree is called a source, and a vertex v with zero outdegree is called a sink.

EXAMPLE 7.2 Consider the graph G in Fig. 7-1(a). We have:

outdeg (A) = 1, outdeg (B) = 4, outdeg (C) = 0, outdeg (D) = 2, indeg (A) = 2, indeg (B) = 2, indeg (C) = 2, indeg (D) = 1.

As expected, the sum of the outdegrees equals the sum of the indegrees, which equals the number 7 of edges. The vertex C is a sink since no edge begins at C. The graph has no sources.

6.12.2 Paths

Let G be a directed graph. The concepts of path, simple path, trail, and cycle carry over from nondirected graphs to the directed graph G except that the directions of the edges must agree with the direction of the path. Specifically:

(i) A (directed) path P in G is an alternating sequence of vertices and directed edges, say,

$$P = (v_0, e_1, v_1, e_2, v_2, ..., e_n, v_n)$$

such that each edge e_i begins at v_{i-1} and ends at v_i. If there is no ambiguity, we denote P by its sequence of vertices or its sequence of edges.

(ii) The length of the path P is n, its number of edges.
(iii) A simple path is a path with distinct vertices. A trail is a path with distinct edges.
(iv) A closed path has the same first and last vertices.
(v) A spanning path contains all the vertices of G.
(vi) A cycle (or circuit) is a closed path with distinct vertices (except the first and last).
(vii) A semipath is the same as a path except the edge e_i may begin at v_{i-1} or v_i and end at the other vertex. Semitrails and semisimple paths are analogously defined.

A vertex v is reachable from a vertex u if there is a path from u to v. If v is reachable from u, then (by eliminating redundant edges) there must be a simple path from u to v.

EXAMPLE 7.3 Consider the graph G in Fig. 7-1(a).

(a) The sequence P_1 = (D, C, B, A) is a semipath but not a path since (C, B) is not an edge; that is, the direction of e_5 = (C, B) does not agree with the direction of P_1.

(b) The sequence P_2 = (D, B, A) is a path from D to A since (D, B) and (B, A) are edges. Thus A is reachable from D.

6.12.3 Connectivity

There are three types of connectivity in a directed graph G:

(i) G is strongly connected or strong if, for any pair of vertices u and v in G, there is a path from u to v and a path from v to u, that is, each is reachable from the other.
(ii) G is unilaterally connected or unilateral if, for any pair of vertices u and v in G, there is a path from u to v or a path from v to u, that is, one of them is reachable from the other.
(iii)G is weakly connected or weak if there is a semipath between any pair of vertices u and v in G.

Let G′ be the (nondirected) graph obtained from a directed graph G by allowing all edges in G to be nondirected. Clearly, G is weakly connected if and only if the graph G′ is connected.

Observe that strongly connected implies unilaterally connected which implies weakly connected. We say that G is strictly unilateral if it is unilateral but not strong, and we say that G is strictly weak if it is weak but not unilateral.

Connectivity can be characterized in terms of spanning paths as follows:

Theorem 7.2: Let G be a finite directed graph. Then:

(i) G is strong if and only if G has a closed spanning path.
(ii) G is unilateral if and only if G has a spanning path.
(iii)G is weak if and only if G has a spanning semipath.

EXAMPLE 7.4 Consider the graph G in Fig. 7-1(a). It is weakly connected since the underlying nondirected graph is connected. There is no path from C to any other vertex, that is, C is a sink, so G is not strongly connected. However, P = (B, A, D, C) is a spanning path, so G is unilaterally connected.

Theorem 7.3: Suppose a finite directed graph G is cycle-free, that is, contains no (directed) cycles. Then G contains a source and a sink.

Proof: Let P = (v_0, v_1,...,v_n) be a simple path of maximum length, which exists since G is finite. Then the last vertex v_n is a sink; otherwise an edge (v_n, u) will either extend P or form a cycle if u = v_i, for some i. Similarly, the first vertex v_0 is a source.

6.13 SEQUENTIAL REPRESENTATION OF DIRECTED GRAPHS

There are two main ways of maintaining a directed graph G in the memory of a computer. One way, called the sequential representation of G, is by means of its adjacency matrix A. The other way, called the linked representation of G, is by means of linked lists of neighbors.

Suppose a graph G has m vertices (nodes) and n edges. We say G is dense if m = O(n^2) and sparse if m = O(n) or even if m = O(n log n). The matrix representation of G is usually used when G is dense, and linked lists are usually used when G is sparse. Regardless of the way one maintains a graph G in memory, the graph G is normally input into the computer by its formal definition, that is, as a collection of vertices and a collection of edges (pairs of vertices).

Remark: In order to avoid special cases of our results, we assume, unless otherwise stated, that m > 1 where m is the number of vertices in our graph G. Therefore, G is not connected if G has no edges.

6.13.1 Digraphs and Relations, Adjacency Matrix

Let G(V, E) be a simple directed graph, that is, a graph without parallel edges. Then E is simply a subset of V × V , and hence E is a relation on V. Conversely, if R is a relation on a set V, then G(V, R) is a simple directed graph.

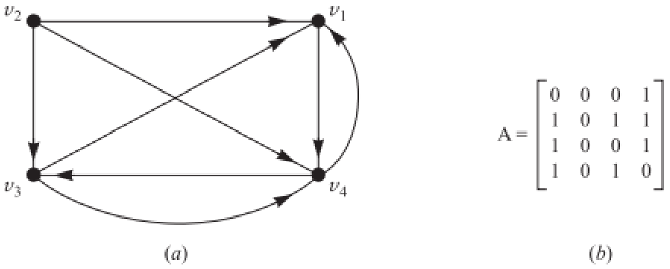
Suppose G is a simple directed graph with m vertices, and suppose the vertices of G have been ordered and are called v_1, v_2,…,v_m. Then the adjacency matrix A = [a_{ij}] of G is the m × m matrix defined as follows:

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

Such a matrix A, which contains entries of only 0 or 1, is called a bit matrix or a Boolean matrix. (Although the adjacency matrix of an undirected graph is symmetric, this is not true here for a directed graph.)

The adjacency matrix A of the graph G does depend on the ordering of the vertices of G. However, the matrices resulting from two different orderings are closely related in that one can be obtained from the other by simply interchanging rows and columns.

EXAMPLE 7.5 Let G be the directed graph in Fig. 7-2(a) with vertices v_1, v_2, v_3, v_4. Then the adjacency matrix A of G appears in Fig. 7-2(b). Note that the number of 1's in A is equal to the number (eight) of edges



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

Fig. 7-2

Consider the powers A, A^2, A^3,… of the adjacency matrix A = [aij] of a graph G. Let

$a_K(i, j)$ = the ij entry in the matrix A^K

Note that $a_1(i, j) = a_{ij}$ gives the number of paths of length 1 from vertex v_i to vertex v_j . One can show that $a_2(i, j)$ gives the number of paths of length 2 from v_i to v_j .

EXAMPLE 7.6 Consider again the graph G and its adjacency matrix A appearing in Fig. 7-2. The powers A^2, A^3, and A^4 of A follow:

$$A^2 = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 2 & 0 & 1 & 2 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 2 \end{bmatrix}, \quad A^3 = \begin{bmatrix} 1 & 0 & 0 & 2 \\ 3 & 0 & 2 & 3 \\ 2 & 0 & 1 & 2 \\ 2 & 0 & 2 & 1 \end{bmatrix}, \quad A^4 = \begin{bmatrix} 2 & 0 & 2 & 1 \\ 5 & 0 & 3 & 5 \\ 3 & 0 & 2 & 3 \\ 3 & 0 & 1 & 4 \end{bmatrix}$$

Observe that $a_2(4, 1) = 1$, so there is a path of length 2 from v_4 to v_1. Also, $a_3(2, 3) = 2$, so there are two paths of length 3 from v_2 to v_3; and $a_4(2, 4) = 5$, so there are five paths of length 4 from v_2 to v_4.

6.13.2 Path Matrix

Let G = G(V , E) be a simple directed graph with m vertices $v_1, v_2,..., v_m$. The path matrix or reachability matrix of G is the m-square matrix $P = [p_{ij}]$ defined as follows:

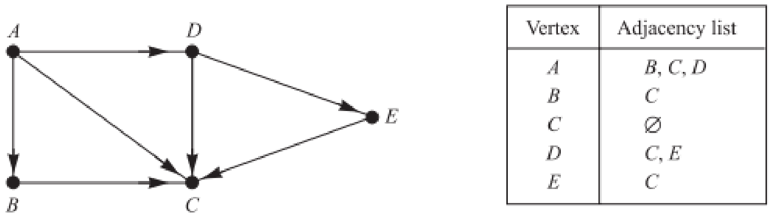
$$a_{ij} = \begin{cases} 1 & \text{if there is an edge } (v_i, v_j) \\ 0 & \text{otherwise} \end{cases}$$

(The path matrix P may be viewed as the transitive closure of the relation E on V .)

Suppose now that there is a path from vertex v_i to vertex v_j in a graph G with m vertices. Then there must be a simple path from v_i to v_j when $v_i \neq v_j$, or there must be a cycle from v_i to v_j when $v_i = v_j$. Since G has m vertices, such a simple path must have length m − 1 or less, or such a cycle must have length m or less.

6.14 LINKED REPRESENTATION OF DIRECTED GRAPHS

Let G be a directed graph with m vertices. Suppose the number of edges of G is O(m) or even O(m log m), that is, suppose G is sparse. Then the adjacency matrix A of G will contain many zeros; hence a great deal of memory space will be wasted. Accordingly, when G is sparse, G is usually represented in memory by some type of linked representation, also called an adjacency structure, which is described below by means of an example.



Vertex	Adjacency list
A	*B*, *C*, *D*
B	*C*
C	∅
D	*C*, *E*
E	*C*

(*a*) Graph *G*

(*b*) Adjacency lists of *G*

Fig. 7-3

Consider the directed graph G in Fig. 7-3(a). Observe that G may be equivalently defined by the table in Fig. 7-3(b), which shows each vertex in G followed by its adjacency list, also called its successors or neighbors. Here the symbol ∅ denotes an empty list. Observe that each edge of G corresponds to a unique vertex in an adjacency list and vice versa. Here G has seven edges and there are seven vertices in the adjacency lists. This table may also be presented in the following compact form where a colon ":" separates a vertex from its list of neighbors, and a semicolon ";" separates the different lists:

G = [A : B,C, D;   B : C;   C : ∅;   D : C, E;   E : C]

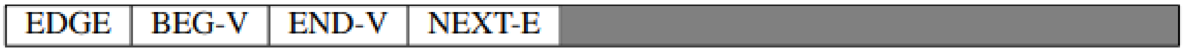
The linked representation of a directed graph G maintains G in memory by using linked lists for its adjacency lists. Specifically, the linked representation will normally contain two files (sets of records), one called the Vertex File and the other called the Edge File, as follows.

(a) Vertex File: The Vertex File will contain the list of vertices of the graph G usually maintained by an array or by a linked list. Each record of the Vertex File will have the form

VERTEX	NEXT-V	PTR	▨

Here VERTEX will be the name of the vertex, NEXT-V points to the next vertex in the list of vertices in the Vertex File, and PTR will point to the first element in the adjacency list of the vertex appearing in the Edge File. The shaded area indicates that there may be other information in the record corresponding to the vertex.

(b) Edge File: The Edge File contains the edges of G and also contains all the adjacency lists of G where each list is maintained in memory by a linked list. Each record of the Edge File will represent a unique edge in G and hence will correspond to a unique vertex in an adjacency list. The record will usually have the form



EDGE	BEG-V	END-V	NEXT-E	

Here:

(1) EDGE will be the name of the edge (if it has a name).

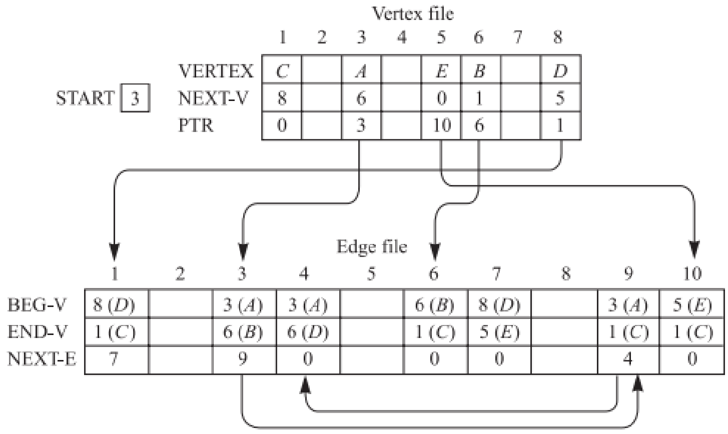
(2) BEG-V- points to location in the Vertex File of the initial (beginning) vertex of the edge.

(3) END-V points to the location in the Vertex File of the terminal (ending) vertex of the edge. The adjacency lists appear in this field.

(4) NEXT-E points to the location in the Edge File of the next vertex in the adjacency list.

We emphasize that the adjacency lists consist of terminal vertices and hence are maintained by the END-V field. The shaded area indicates that there may be other information in the record corresponding to the edge. We note that the order of the vertices in any adjacency list does depend on the order in which the edges (pairs of vertices) appear in the input.

Figure 7-4 shows how the graph G in Fig. 7-3(a) may appear in memory. Here the vertices of G are maintained in memory by a linked list using the variable START to point to the first vertex. (Alternatively, one could use a linear array for the list of vertices, and then NEXT-V would not be required.) The choice of eight locations for the Vertex File and 10 locations for the Edge File is arbitrary. The additional space in the files will be used if additional vertices or edges are inserted in the graph. Figure 7-4 also shows, with arrows, the adjacency list [B, C, D] of the vertex A.



Vertex file

	1	2	3	4	5	6	7	8
VERTEX	*C*		*A*		*E*	*B*		*D*
NEXT-V	8		6		0	1		5
PTR	0		3		10	6		1

Edge file

	1	2	3	4	5	6	7	8	9	10
BEG-V	8 (*D*)		3 (*A*)	3 (*A*)		6 (*B*)	8 (*D*)		3 (*A*)	5 (*E*)
END-V	1 (*C*)		6 (*B*)	6 (*D*)		1 (*C*)	5 (*E*)		1 (*C*)	1 (*C*)
NEXT-E	7		9	0		0	0		4	0

Fig. 7-4

8.1 BOOLEAN ALGEBRA

8.2 BASIC DEFINITIONS

Let B be a nonempty set with two binary operations + and ∗, a unary operation ′, and two distinct elements 0 and 1. Then B is called a Boolean algebra if the following axioms hold where a, b, c are any elements in B:

[B1] Commutative laws:

(1a) $a + b = b + a$ (1b) $a * b = b * a$

[B2] Distributive laws:

(2a) $a + (b * c) = (a + b) * (a + c)$ (2b) $a * (b + c) = (a * b) + (a * c)$

[B3] Identity laws:

(3a) $a + 0 = a$ (3b) $a * 1 = a$

[B4] Complement laws:

(4a) $a + a' = 1$ (4b) $a * a' = 0$

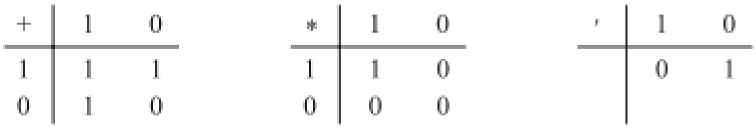
We will sometimes designate a Boolean algebra by $\langle B, +, *, ' \; 0, 1\rangle$ when we want to emphasize its six parts. We say 0 is the zero element, l, is the unit element, and a' is the complement of a. We will usually drop the symbol $*$ and use juxtaposition instead.

The operations $+$, $*$, and $'$ are called sum, product, and complement, respectively. We adopt the usual convention that, unless we are guided by parentheses, $'$ has precedence over $*$, and $*$ has precedence over $+$. For example,

$a + b * c$ means $a + (b * c)$ and not $(a + b) * c$; $a * b'$ means $a * (b')$ and not $(a * b)'$

EXAMPLE 8.1

Let $B = \{0, 1\}$, the set of bits (binary digits), with the binary operations of $+$ and $*$ and the unary operation $'$ defined by Fig. 8-1. Then B is a Boolean algebra. (Note simply changes the bit, i.e., 1 = 0 and 0 = 1.)



+	1	0
1	1	1
0	1	0

*	1	0
1	1	0
0	0	0

'	1	0
	0	1

Fig. 8-1

8.3 DUALITY

The dual of any statement in a Boolean algebra B is the statement obtained by interchanging the operations

$+$ and $*$, and interchanging their identity elements 0 and 1 in the original statement. For example, the dual of

$$(1 + a) * (b + 0) = b \quad \text{is} \quad (0 * a) + (b * 1) = b$$

8.4 BASIC THEOREMS

Using the axioms [B1] through [B4], we prove the following theorem.

Theorem 8.1: Let a, b, c be any elements in a Boolean algebra B.

(i) Idempotent laws:

(5a) $a + a = a$ (5b) $a * a = a$

(ii) Boundedness laws:

(6a) $a + 1 = 1$ (6b) $a * 0 = 0$

(iii) Absorption laws:

(7a) $a + (a * b) = a$ (7b) $a * (a + b) = a$

(iv) Associative laws:

(8a) $(a + b) + c = a + (b + c)$ (8b) $(a * b) * c = a * (b * c)$

(v) (Involution law) $(a')' = a$.

(9a) $0' = 1$. (9b) $1' = 0$.

(vi) (DeMorgan's laws):

(10a) $(a + b)' = a' * b'$. (10b) $(a * b)' = a' + b'$.

8.5 SUM-OF-PRODUCTS FORM FOR SETS

This section motivates the concept of the sum-of-products form in Boolean algebra by an example of set theory. Consider the Venn diagram in Fig. 8-2 of three sets A, B, and C. Observe that these sets partition the rectangle (universal set) into eight numbered sets which can be represented as follows:

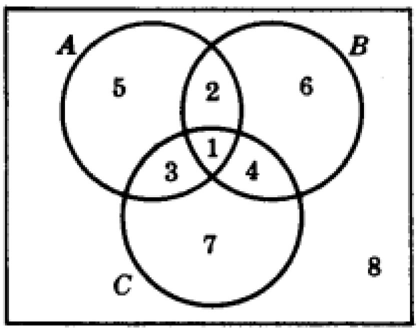


Fig. 8-2

(1) $A \cap B \cap C$

(2) $A \cap B \cap C^c$

(3) $A \cap B^c \cap C$

(4) $A^c \cap B \cap C$

(5) $A \cap B^c \cap C^c$

(6) $A^c \cap B \cap C^c$

(7) $A^c \cap B^c \cap C$

(8) $A^c \cap B^c \cap C^c$

Each of these eight sets is of the form $A^* \cap B^* \cap C^*$ where:

$$A^* = A \text{ or } A^c,\ B^* = B \text{ or } B^c,\ C^* = C \text{ or } C^c$$

Consider any nonempty set expression E involving the sets A, B, and C, say,

$$E = [(A \cap B^c)^c \cup (A^c \cap C^c)] \cap [(B^c \cup C)^c \cap (A \cup C^c)]$$

Then E will represent some area in Fig. 8-2 and hence will uniquely equal the union of one or more of the eight sets. Suppose we now interpret a union as a sum and an intersection as a product.

Then the above eight sets are products, and the unique representation of E will be a sum (union) of products. This unique representation of E is the same as the complete sum-of-products expansion in Boolean algebras which we discuss below.

8.6 SUM-OF-PRODUCTS FORM FOR BOOLEAN ALGEBRAS

Consider a set of variables (or letters or symbols), say x_1, x_2,...,x_n. A Boolean expression E in these variables, sometimes written E(x_1,...,x_n), is any variable or any expression built up from the variables using the Boolean operations +, ∗, and ′. (Naturally, the expression E must be well-formed, that is, where + and ∗ are used as binary operations, and ′ is used as a unary operation.) For example,

$$E_1 = (x + y'z)' + (xyz' + x'y)' \text{ and } E_2 = ((xy'z' + y)' + x'z)'$$

are Boolean expressions in x, y, and z.

A literal is a variable or complemented variable, such as x ′ , y, y ′ , and so on. A fundamental product is a literal or a product of two or more literals in which no two literals involve the same variable. Thus

$$xz',\ xy'z,\ x,\ y',\ x'yz$$

are fundamental products, but $xyx'z$ and xyzy are not. Note that any product of literals can be reduced to either 0 or a fundamental product, e.g., $xyx'z = 0$ since $xx' = 0$ (complement law), and xyzy = xyz since yy = y (idempotent law).

A fundamental product P_1 is said to be contained in (or included in) another fundamental product P_2 if the literals of P_1 are also literals of P_2. For example, $x'z$ is contained in $xy'z$, since x' is not a literal of $xy'z$. Observe that if P_1 is contained in P_2, say $P_2 = P_1 * Q$, then, by the absorption law,

$$P_1 + P_2 = P_1 + P_1 * Q = P_1$$

Thus, for instance, $x'z + x'yz = x'z$.

Definition 8.1: A Boolean expression E is called a sum-of-products expression if E is a fundamental product or the sum of two or more fundamental products none of which is contained in another.

Definition 8.2: Let E be any Boolean expression. A sum-of-products form of E is an equivalent Boolean sum-of-products expression.

EXAMPLE 8.2 Consider the expressions

$$E_1 = xz' + y'z + xyz' \text{ and } E_2 = xz' + x'yz' + xy'z$$

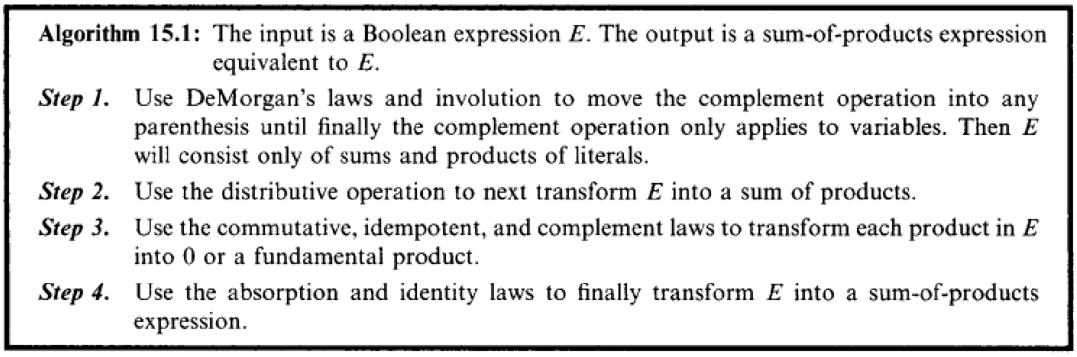
Although the first expression E_1 is a sum of products, it is not a sum-of-products expression. Specifically, the product xz ′ is contained in the product xyz ′. However, by the absorption law, E_1 can be expressed as

$$E_1 = xz' + y'z + xyz' = xz' + xyz' + y'z = xz' + y'z$$

This yields a sum-of-products form for E$_1$. The second expression E$_2$ is already a sum-of-products expression.

Algorithm for Finding Sum-of-Products Forms

Figure 8-3 gives a four-step algorithm which uses the Boolean algebra laws to transform any Boolean expression into an equivalent sum-of-products expression.



Algorithm 15.1: The input is a Boolean expression E. The output is a sum-of-products expression equivalent to E.

Step 1. Use DeMorgan's laws and involution to move the complement operation into any parenthesis until finally the complement operation only applies to variables. Then E will consist only of sums and products of literals.

Step 2. Use the distributive operation to next transform E into a sum of products.

Step 3. Use the commutative, idempotent, and complement laws to transform each product in E into 0 or a fundamental product.

Step 4. Use the absorption and identity laws to finally transform E into a sum-of-products expression.

Fig. 8-3

EXAMPLE 8.3 Suppose Algorithm 15.1 is applied to the following Boolean expression:

$$E = ((xy)'z)'\big((x' + z)(y' + z')\big)'$$

Step 1. Using DeMorgan's laws and involution, we obtain

$$E = ((xy)'' + z')((x' + z)' + (y' + z')') = (xy + z')(xz' + yz)$$

E now consists only of sums and products of literals.

Step 2. Using the distributive laws, we obtain

$$E = xyxz' + xyyz + xz'z' + yzz'$$

E now is a sum of products.

Step 3. Using the commutative, idempotent, and complement laws, we obtain

$$E = xyz' + xyz + xz' + 0$$

Each term in E is a fundamental product or 0.

Step 4. The product xz' is contained in xyz'; hence, by the absorption law,

$$xz' + (xz'y) = xz'$$

Thus we may delete xyz from the sum. Also, by the identity law for 0, we may delete 0 from the sum. Accordingly,

$$\text{E} = xyz + xz'$$

E is now represented by a sum-of-products expression.

9.1 LOGIC GATES AND CIRCUITS

Logic circuits (also called logic networks) are structures which are built up from certain elementary circuits called logic gates. Each logic circuit may be viewed as a machine L which contains one or more input devices and exactly one output device. Each input device in L sends a signal, specifically, a bit (binary digit),

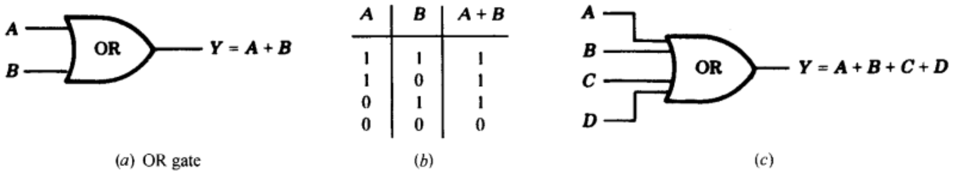
0 or 1

to the circuit L, and L processes the set of bits to yield an output bit. Accordingly, an n-bit sequence may be assigned to each input device, and L processes the input sequences one bit at a time to produce an n-bit output sequence. First we define the logic gates, and then we investigate the logic circuits.

9.2 Logic Gates

There are three basic logic gates which are described below. We adopt the convention that the lines entering the gate symbol from the left are input lines and the single line on the right is the output line.

(a) OR Gate: Figure 9-4(a) shows an OR gate with inputs A and B and output Y = A + B where "addition" is defined by the "truth table" in Fig. 9-4(b). Thus the output Y = 0 only when inputs A = 0 and B = 0. Such an OR gate may, have more than two inputs. Figure 9-4(c) shows an OR gate with four inputs, A, B, C, D, and output Y = A + B + C + D. The output Y = 0 if and only if all the inputs are 0.



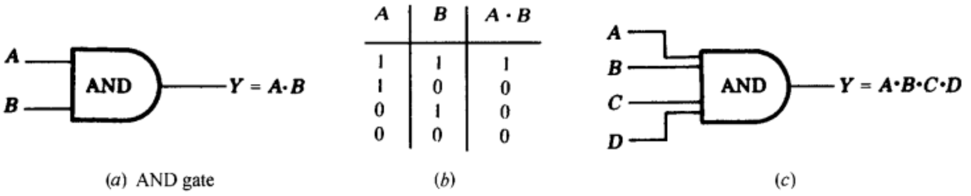
A	B	A + B
1	1	1
1	0	1
0	1	1
0	0	0

(a) OR gate (b) (c)

Fig. 9-4

The OR gate only yields 0 when all input bits are 0.

(b) AND Gate: Figure 9-5(a) shows an AND gate with inputs A and B and output Y = A · B (or simply Y = AB) where "multiplication" is defined by the "truth table" in Fig. 9-5(b). Thus the output Y = 1 when inputs A = 1 and B = 1; otherwise Y = 0. Such an AND gate may have more than two inputs. Figure 9-5(c) shows an AND gate with four inputs, A, B, C, D, and output Y = A · B · C · D. The output Y = 1 if and only if all the inputs are 1.

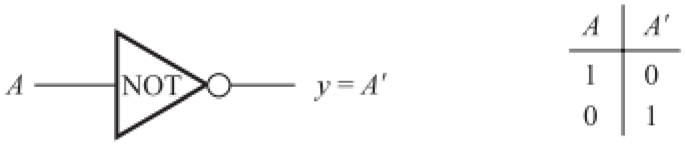


A	B	A · B
1	1	1
1	0	0
0	1	0
0	0	0

(a) AND gate (b) (c)

Fig. 9-5

(c) NOT Gate: Figure 9-6(a) shows a NOT gate, also called an inverter, with input A and output Y = A where "inversion," denoted by the prime, is defined by the "truth table" in Fig. 9-6(b). The value of the output Y = A is the opposite of the input A; that is, A = 1 when A = 0 and A = 0 when A = 1. We emphasize that a NOT gate can have only one input, whereas the OR and AND gates may have two or more inputs.



A	A'
1	0
0	1

(a) NOT gate

(b)

Fig. 9-6

9.3 Logic Circuits

A logic circuit L is a well-formed structure whose elementary components are the above OR, AND, and NOT gates. Figure 9-7 is an example of a logic circuit with inputs A, B, C and output Y . A dot indicates a place where the input line splits so that its bit signal is sent in more than one direction. (Frequently, for notational convenience, we may omit the word from the interior of the gate symbol.) Working from left to right, we express Y in terms of the inputs A, B, C as follows. The output of the AND gate is A · B, which is then negated to yield $(A \cdot B)'$. The output of the lower OR gate is A + C, which is then negated to yield $(A + B)'$. The output of the OR gate on the right, with inputs $(A \cdot B)'$ and $(A' + C)'$, gives us our desired representation, that is,

$$Y = (A \cdot B)' + (A' + C)'$$

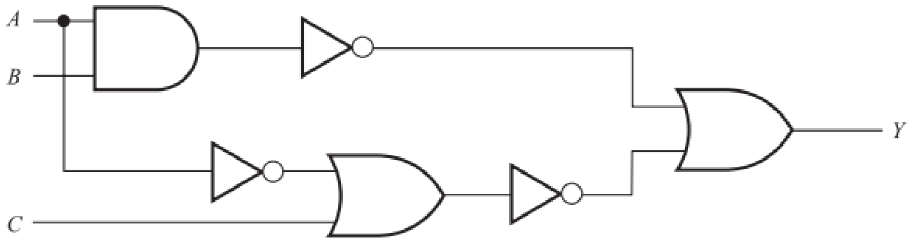


Fig. 9-7

9.4 Logic Circuits as a Boolean Algebra

Observe that the truth tables for the OR, AND, and NOT gates are respectively identical to the truth tables for the propositions p ∨ q (disjunction, "p or q"), p ∧ q (conjunction, "p and q"), and ¬p (negation, "not p"). The only difference is that 1 and 0 are used instead of T and F. Thus the logic circuits satisfy the same laws as do propositions and hence they form a Boolean algebra. We state this result formally.

Theorem 9.1: Logic circuits form a Boolean Algebra. Accordingly, all terms used with Boolean algebras, such as, complements, literals, fundamental products, minterms, sum-of-products, and complete sum-of-products, may also be used with our logic circuits.

9.4.1 AND-OR Circuits

The logic circuit L which corresponds to a Boolean sum-of-products expression is called an AND-OR circuit. Such a circuit L has several inputs, where:

(1) Some of the inputs or their complements are fed into each AND gate.

(2) The outputs of all the AND gates are fed into a single OR gate.

(3) The output of the OR gate is the output for the circuit L.

The following illustrates this type of a logic circuit.

EXAMPLE 9.4 Figure 9-8 is a typical AND-OR circuit with three inputs, A, B, C and output Y. We can easily express Y as a Boolean expression in the inputs A, B, C as follows. First we find the output of each AND gate:

(a) The inputs of the first AND gate are A, B, C; hence $A \cdot B \cdot C$ is the output.

(b) The inputs of the second AND gate are A, B, C; hence $A \cdot B \cdot C$ is the output.

(c) The inputs of the third AND gate are A and B; hence $A \cdot B$ is the output.

Then the sum of the outputs of the AND gates is the output of the OR gate, which is the output Y of the circuit. Thus:

$$Y = A \cdot B \cdot C + A \cdot B' \cdot C + A' \cdot B$$

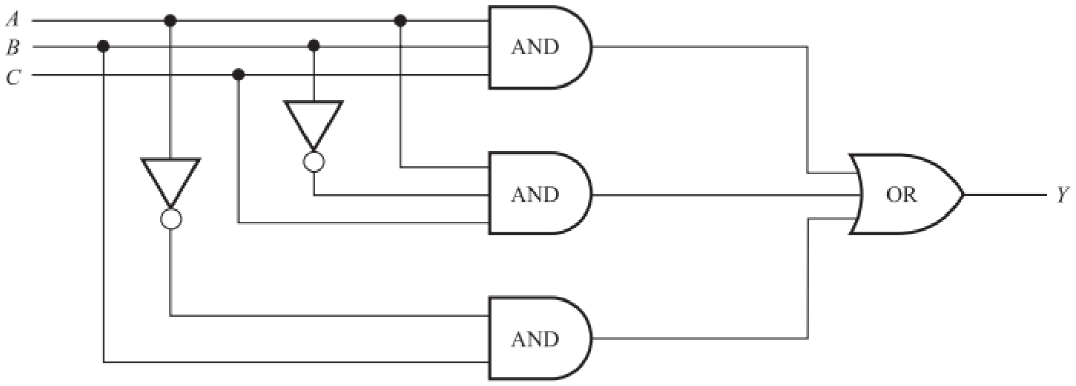


Fig. 9-8

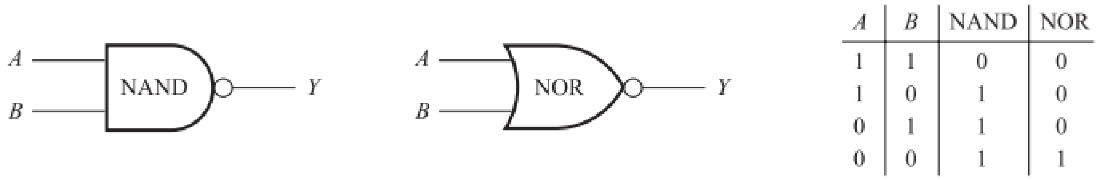
9.4.2 NAND and NOR Gates

There are two additional gates which are equivalent to combinations of the above basic gates.

(a) A NAND gate, pictured in Fig. 9-9(a), is equivalent to an AND gate followed by a NOT gate.

(b) A NOR gate, pictured in Fig. 9-9(b), is equivalent to an OR gate followed by a NOT gate.

The truth tables for these gates (using two inputs A and B) appear in Fig. 9-9(c). The NAND and NOR gates can actually have two or more inputs just like the corresponding AND and OR gates. Furthermore, the output of a NAND gate is 0 if and only if all the inputs are 1, and the output of a NOR gate is 1 if and only if all the inputs are 0.



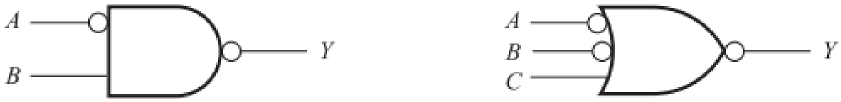
(a) NAND gate (b) NOR gate (c)

A	B	NAND	NOR
1	1	0	0
1	0	1	0
0	1	1	0
0	0	1	1

Fig. 9-9

Observe that the only difference between the AND and NAND gates and between the OR and NOR gates is that the NAND and NOR gates are each followed by a circle. Some texts also use such a small circle to indicate a complement before a gate. For example, the Boolean expressions corresponding to two logic circuits in Fig. 9-10 are as follows:

(a) $Y = (A'B)'$, (b) $Y = (A' + B' + C)'$



(a) (b)

Fig. 9-10

9.5 KARNAUGH MAPS

Karnaugh maps, where minterms involving the same variables are represented by squares, are pictorial devices for finding minimal forms for Boolean expressions involving at most six variables. We will only treat the cases of two, three, and four variables. In the context of Karnaugh maps, we will sometimes use the terms “squares” and “minterm” interchangeably. Recall that a minterm is a fundamental product which involves all the variables, and that a complete sum-of-products expression is a sum of minterms.

First we need to define the notion of adjacent products. Two fundamental products P1 and P2 are said to be adjacent if P1 and P2 have the same variables and if they differ in exactly one literal. Thus there must be an uncomplemented variable in one product and complemented in the other. In particular, the sum of two such adjacent products will be a fundamental product with one less literal.

EXAMPLE 9.5 Find the sum of adjacent products P1 and P2 where:

(a) $P_1 = xyz'$ and $P_2 = xy'z'$

$$P_1 + P_2 = xyz' + xy'z' = xz'(y + y') = xz'(1) = xz'$$

(b) $P_1 = x'yzt$ and $P_2 = x'yz't$.

$$P_1 + P_2 = x'yzt + x'yz't = x'yt(z + z') = x'yt(1) = x'yt$$

(c) $P_1 = x'yzt$ and $P_2 = xyz't$.

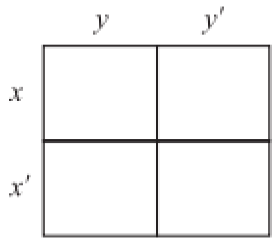
Here P1 and P2 are not adjacent since they differ in two literals. In particular,

$$P_1 + P_2 = x'yzt + xyz't = (x' + x)y(z + z')t = (1)y(1)t = yt$$

(d) $P_1 = xyz'$ and $P_2 = xyzt$.

Here P_1 and P_2 are not adjacent since they have different variables. Thus, in particular, they will not appear as squares in the same Karnaugh map.

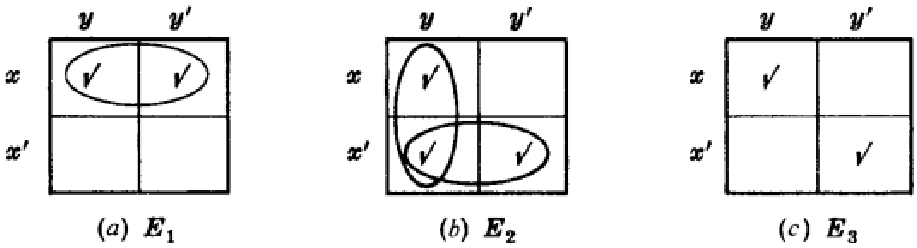
Two Variables Karnaugh map



EXAMPLE 9.6 Find the minimal sum-of-products form for each of the following complete sum-of-products Boolean expressions:

(a) $E_1 = xy + xy'$; (b) $E_2 = xy + x'y + x'y'$; (c) $E_3 = xy + x'y'$

This can be solved by using Karnaugh maps as follows:

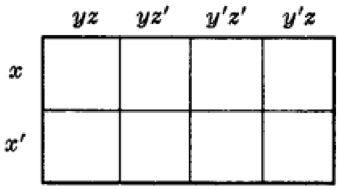


(a) E_1 (b) E_2 (c) E_3

Fig. 9-11

$E_1 = x$; $E_2 = x' + y$; $E_3 = xy + x'y'$

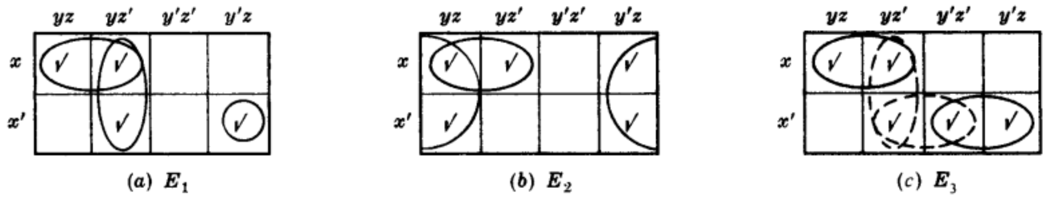
Three Variables Karnaugh map



EXAMPLE 9.7 Find the minimal sum-of-products form for each of the following complete sum-of-products Boolean expressions:

(a) $E_1 = xyz + xyz' + x'yz' + x'y'z$.
(b) $E_2 = xyz + xyz' + xy'z + x'yz + x'y'z$.
(c) $E_3 = xyz + xyz' + x'yz' + x'y'z' + x'y'z$.

This can be solved by using Karnaugh maps as follows:



(a) E_1 (b) E_2 (c) E_3

Fig. 9-12

$E_1 = xy + yz' + x'y'z$

$E_2 = xy + z$

$E_3 = xy + yz' + x'z' + x'y'$

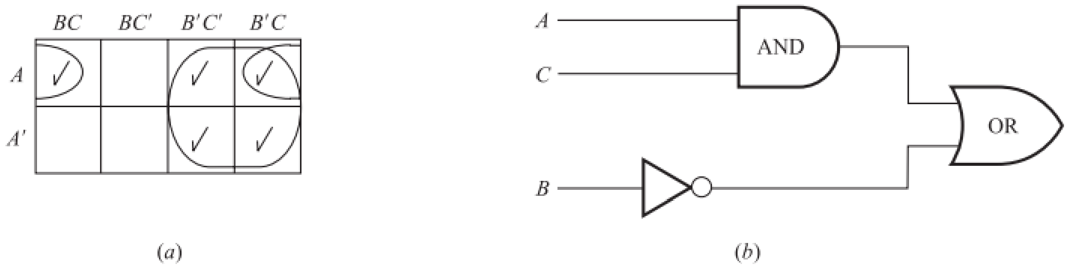
EXAMPLE 9.8 Design a three-input minimal AND-OR circuit L with the following truth table:

T = [A, B, C; L]=[00001111, 00110011, 01010101; 11001101]

From the truth table we can read off the complete sum-of-products form for L (as in Example 9.4):

$$L = A'B'C' + A'B'C + AB'C' + AB'C + ABC$$

The associated Karnaugh map is shown in Fig. 9-12(a). Observe that L = B' + AC is a minimal sum for L. Figure 9-12(b) gives the corresponding minimal AND-OR circuit for L.



(a) (b)

Fig. 9-12

Four Variables Karnaugh map

	zt	zt'	$z't'$	$z't$
xy				
xy'				
$x'y'$				
$x'y$				

EXAMPLE 9.9 Find the fundamental product P represented by the basic rectangle in the Karnaugh maps shown in Fig. 9-13.

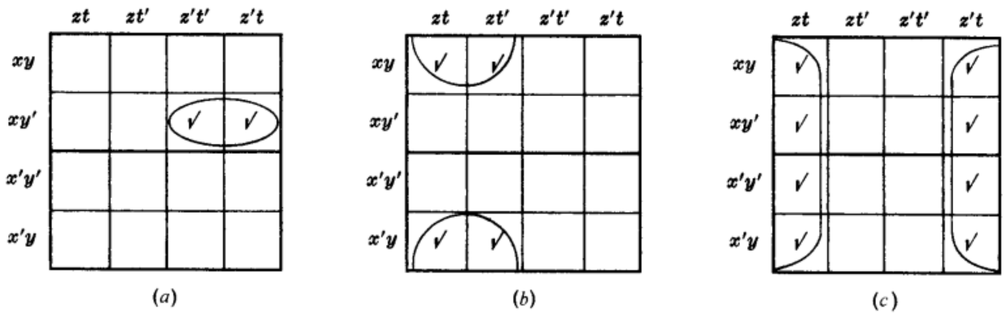


Fig. 9-13

(a) xy', and z' appear in both squares; hence P = $xy'z'$

(b) Only y and z appear in all four squares; hence P = yz.

(c) Only t appears in all eight squares; hence P = t.

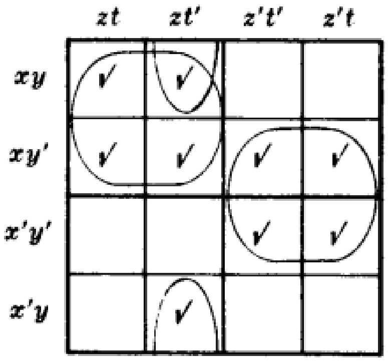
EXAMPLE 9.10 Use a Karnaugh map to find a minimal sum-of-products form for

$$E = xy' + xyz + x'y'z' + x'yzt'$$

$$E = xy'(z' + z)$$

$$E = xy'z'(t' + t) + xy'z(t' + t)$$

$$E = xy'z't' + xy'z't + xy'zt' + xy'zt + xyzt' + xyzt + x'y'z't' + x'y'z't + x'yzt'$$



Hence, P $= yzt' + xz + y'z'$

10.1 VECTORS AND MATRICES

Data is frequently arranged in arrays, that is, sets whose elements are indexed by one or more subscripts. If the data consists of numbers, then a one-dimensional array is called a vector and a two-dimensional array is called a matrix (where the dimension denotes the number of subscripts).

10.2 VECTORS

By a vector u, we mean a list of numbers, say, a_1, a_2,...,a_n. Such a vector is denoted by

$u = (a_1, a_2,...,a_n)$

The numbers a_i are called the components or entries of u. If all the $a_i = 0$, then u is called the zero vector. Two such vectors, u and v, are equal, written u = v, if they have the same number of components and corresponding components are equal.

EXAMPLE 10.1

(a) The following are vectors where the first two have two components and the last two have three components:

(3, −4), (6, 8), (0, 0, 0), (2, 3, 4)

The third vector is the zero vector with three components.

(b) Although the vectors (1, 2, 3) and (2, 3, 1) contain the same numbers, they are not equal since corresponding components are not equal.

10.3 Vector Operations

Consider two arbitrary vectors u and v with the same number of components, say

$u = (a_1, a_2,...,a_n)$ and $v = (b_1, b_2,...,b_n)$

The sum of u and v, written u + v, is the vector obtained by adding corresponding components from u and v; that is,

$u + v = (a_1 + b_1, a_2 + b_2,...,a_n + b_n)$

The scalar product or, simply, product, of a scalar k and the vector u, written ku, is the vector obtained by multiplying each component of u by k; that is,

$ku = (ka_1, ka_2, \ldots , ka_n)$

We also define

$-u = -1(u)$ and $u - v = u + (-v)$

and we let 0 denote the zero vector. The vector −u is called the negative of the vector u.

The dot product or inner product of the above vectors u and v is denoted and defined by

$u \cdot v = a_1b_1 + a_2b_2 + \cdots + a_nb_n$

The norm or length of the vector u is denoted and defined by

$$||u|| = \sqrt{u \cdot u} = \sqrt{a_1^2 + a_2^2 + \cdots + a_n^2}$$

We note that $||u|| = 0$ if and only if u = 0; otherwise $||u|| > 0$.

EXAMPLE 10.2 Let u = (2, 3, −4) and v = (1, −5, 8). Then

$u + v = (2 + 1, 3 - 5, -4 + 8) = (3, -2, 4)$

$5u = (5 \cdot 2, 5 \cdot 3, 5 \cdot (-4)) = (10, 15, -20)$

$-v = -1 \cdot (1, -5, 8) = (-1, 5, -8)$

$2u - 3v = (4, 6, -8) + (-3, 15, -24) = (1, 21, -32)$

$u \cdot v = 2 \cdot 1 + 3 \cdot (-5) + (-4) \cdot 8 = 2 - 15 - 32 = -45$

$||u|| = \sqrt{2^2 + 3^2 + (-4)^2} = \sqrt{4 + 9 + 16} = \sqrt{29}$

Vectors under the operations of vector addition and scalar multiplication have various properties, e.g.,

$k(u + v) = ku + kv$

where k is a scalar and u and v are vectors.

10.4 Column Vectors

Sometimes a list of numbers is written vertically rather than horizontally, and the list is called a column vector. In this context, the above horizontally written vectors are called row vectors. The above operations for row vectors are defined analogously for column vectors.

10.5 MATRICES

A matrix A is a rectangular array of numbers usually presented in the form

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots\cdots & \cdots\cdots & \cdots\cdots & \cdots. \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

The m horizontal lists of numbers are called the rows of A, and the n vertical lists of numbers its columns. Thus the element a_{ij} , called the ij entry, appears in row i and column j. We frequently denote such a matrix by simply writing A = $[a_{ij}]$.

A matrix with m rows and n columns is called an m by n matrix, written m × n. The pair of numbers m and n is called the size of the matrix. Two matrices A and B are equal, written A = B, if they have the same size and if corresponding elements are equal. Thus, the equality of two m × n matrices is equivalent to a system of *mn* equalities, one for each corresponding pair of elements.

A matrix with only one row is called a row matrix or row vector, and a matrix with only one column is called a column matrix or column vector. A matrix whose entries are all zero is called a zero matrix and will usually be denoted by 0.

EXAMPLE 10.3

(a) The rectangular array $A = \begin{bmatrix} 1 & -4 & 5 \\ 0 & 3 & -2 \end{bmatrix}$ is a 2 × 3 matrix. Its rows are [1, −4, 5] and [0, 3, −2], and its columns are $\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -4 \\ 3 \end{bmatrix}, \begin{bmatrix} 5 \\ -2 \end{bmatrix}$.

(b) The 2 × 4 zero matrix is the matrix 0 =$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$.

(c) Suppose $\begin{bmatrix} x+y & 2z+t \\ x-y & z-t \end{bmatrix} = \begin{bmatrix} 3 & 7 \\ 1 & 5 \end{bmatrix}$

Then the four corresponding entries must be equal. That is,

$$x + y = 3,\ x - y = 1,\ 2z + t = 7,\ z - t = 5$$

The solution of the system of equations is

$$x = 2,\ y = 1,\ z = 4,\ t = -1$$

10.6 MATRIX ADDITION AND SCALAR MULTIPLICATION

Let A = $[a_{ij}]$ and B = $[b_{ij}]$ be two matrices of the same size, say, m × n matrices. The sum of A and B, written A + B, is the matrix obtained by adding corresponding elements from A and B. The (scalar) product of the matrix A by a scalar k, written kA, is the matrix obtained by multiplying each element of A by k. These operations are pictured in Fig. 10-1.

$$A+B=\begin{bmatrix} a_{11}+b_{11} & a_{12}+b_{12} & \cdots & a_{1n}+b_{1n} \\ a_{21}+b_{21} & a_{22}+b_{22} & \cdots & a_{2n}+b_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1}+b_{m1} & a_{m2}+b_{m2} & \cdots & a_{mn}+b_{mn} \end{bmatrix} \quad \text{and} \quad kA=\begin{bmatrix} ka_{11} & ka_{12} & \cdots & ka_{1n} \\ ka_{21} & ka_{22} & \cdots & ka_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ ka_{m1} & ka_{m2} & \cdots & ka_{mn} \end{bmatrix}$$

Fig. 10-1

Observe that A + B and kA are also m × n matrices. We also define

$$-A = (-1)A \text{ and } A - B = A + (-B)$$

The matrix −A is called the negative of A. The sum of matrices with different sizes is not defined.

EXAMPLE 7.4 Let $A = \begin{bmatrix} 1 & -2 & 3 \\ 0 & 4 & 5 \end{bmatrix}$ and $B = \begin{bmatrix} 4 & 6 & 8 \\ 1 & -3 & -7 \end{bmatrix}$. Then

$$A+B = \begin{bmatrix} 1+4 & -2+6 & 3+8 \\ 0+1 & 4+(-3) & 5+(-7) \end{bmatrix} = \begin{bmatrix} 5 & 4 & 11 \\ 1 & 1 & -2 \end{bmatrix}$$

$$3A = \begin{bmatrix} 3(1) & 3(-2) & 3(3) \\ 3(0) & 3(4) & 3(5) \end{bmatrix} = \begin{bmatrix} 3 & -6 & 9 \\ 0 & 12 & 15 \end{bmatrix}$$

$$2A - 3B = \begin{bmatrix} 2 & -4 & 6 \\ 0 & 8 & 10 \end{bmatrix} + \begin{bmatrix} -12 & -18 & -24 \\ -3 & 9 & 21 \end{bmatrix} = \begin{bmatrix} -10 & -22 & -18 \\ -3 & 17 & 31 \end{bmatrix}$$

Matrices under matrix addition and scalar multiplication have the following properties.

Theorem 10.1: Let A, B, C be matrices with the same size, and let k be scalars. Then:

(i) (A + B) + C = A + (B + C) (v) k(A + B) = kA + kB

(ii) A + 0 = 0 + A (vi) (k + k')A = kA + k'A

(iii) A + (−A) = (−A) + 0 = A (vii) (kk')A = k(k'A)

(iv) A + B = B + A (viii) 1A = A

Note first that the 0 in (ii) and (iii) refers to the zero matrix. Also, by (i) and (iv), any sum of matrices

$$A_1 + A_2 + \cdots + A_n$$

requires no parentheses, and the sum does not depend on the order of the matrices. Furthermore, using (vi) and (viii), we also have

$$A + A = 2A, \qquad A + A + A = 3A, \cdots$$

Lastly, since n-component vectors may be identified with either 1 × n or n × 1 matrices, Theorem 10.1 also holds for vectors under vector addition and scalar multiplication.

10.7 MATRIX MULTIPLICATION

The product of matrices A and B, written AB, is somewhat complicated. For this reason, we first begin with a special case. The product AB of a row matrix A = [a_i] and a column matrix B = [b_i] with the same number of elements is defined as follows:

$$AB = [a_1, a_2, \dots, a_n]\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n = \sum_{k=1}^{n} a_k b_k$$

That is, AB is obtained by multiplying corresponding entries in A and B and then adding all the products. We emphasize that AB is a scalar (or a 1 × 1 matrix). The product AB is not defined when A and B have different numbers of elements.

EXAMPLE 7.5

$$\text{(a)}\ [7, \quad -4, \quad 5]\begin{bmatrix} 3 \\ 2 \\ -1 \end{bmatrix} = 7(3) + (-4)(2) + 5(-1) = 21 - 8 - 5 = 8$$

$$\text{(b)}\ [6, \quad -1, \quad 8 \quad 3]\begin{bmatrix} 4 \\ -9 \\ -2 \\ 5 \end{bmatrix} = 24 + 9 - 16 + 15 = 32$$

We are now ready to define matrix multiplication in general.

Definition 10.1: Let A = [a_{ik}] and B = [b_{kj}] be matrices such that the number of columns of A is equal to the number of rows of B, say, A is an m×p matrix and B is a p ×n matrix. Then the product AB is the m×n matrix C = [c_{ij}] whose ij -entry is obtained by multiplying the ith row of A by the jth column of B, that is,

$$c_{ij} = a_{i1} b_{1j} + a_{i2} b_{2j} + \cdots + a_{ip} b_{pj} = \sum_{k=1}^{p} a_{ik} b_{kj}$$

The product AB is pictured in Fig. 10-2.

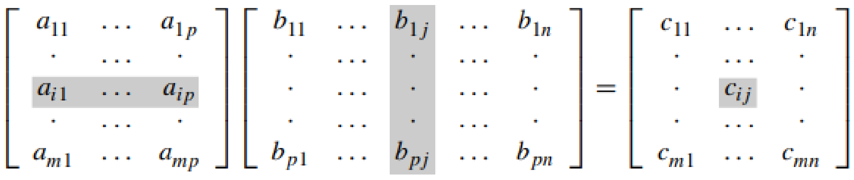


Fig. 10-2

We emphasize that the product AB is not defined if A is an m×p matrix and B is a q ×n matrix where p ≠ q.

EXAMPLE 10.6

(a) Find AB where A =$\begin{bmatrix}1 & 3\\ 2 & -1\end{bmatrix}$ and B=$\begin{bmatrix}2 & 0 & -4\\ 5 & -2 & 6\end{bmatrix}$.

Since A is 2 × 2 and B is 2 × 3, the product AB is defined and AB is a 2 × 3 matrix. To obtain the first row of the product matrix AB, multiply the first row (1, 3) of A times each column of B,

$$\begin{bmatrix}2\\ 5\end{bmatrix}, \quad \begin{bmatrix}0\\ -2\end{bmatrix}, \quad \begin{bmatrix}-4\\ 6\end{bmatrix}$$

respectively. That is,

$$\text{AB} = \begin{bmatrix}2 + 15 & 0 - 6 & -4 + 18\end{bmatrix} = \begin{bmatrix}17 & -6 & 14\end{bmatrix}$$

To obtain the second row of the product AB, multiply the second row (2, −1) of A times each column of B, respectively. Thus

$$AB = \begin{bmatrix}17 & -6 & 14\\ 4-5 & 0+2 & -8-6\end{bmatrix} = \begin{bmatrix}17 & -6 & 14\\ -1 & 2 & -14\end{bmatrix}$$

(b) Suppose A =$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$ and B =$\begin{bmatrix}5 & 6\\ 0 & -2\end{bmatrix}$. Then

$AB = \begin{bmatrix}15+0 & 6-4\\ 15+0 & 18-8\end{bmatrix}$=$\begin{bmatrix}5 & 2\\ 15 & 10\end{bmatrix}$ and $BA = \begin{bmatrix}5+18 & 10+24\\ 0-6 & 0-8\end{bmatrix}$=$\begin{bmatrix}23 & 34\\ -6 & -8\end{bmatrix}$

The above Example 10.6(b) shows that matrix multiplication is not commutative, that is, that the products AB and BA of matrices need not be equal.

Matrix multiplication does, however, satisfy the following properties:

Theorem 10.2: Let A, B, C be matrices. Then, whenever the products and sums are defined:

(i) (AB)C = A(BC) (Associative Law).

(ii) A(B + C) = AB + AC (Left Distributive Law).

(iii) (B + C)A = BA + CA (Right Distributive Law).

(iv) k(AB) = (kA)B = A(kB) where k is a scalar.

10.7.1 Matrix Multiplication and Systems of Linear Equations

Any system S of linear equations is equivalent to the matrix equation

AX = B

where A is the matrix consisting of the coefficients, X is the column vector of unknowns, and B is the column vector of constants. (Here equivalent means that any solution of the system S is a solution to the matrix equation AX = B, and vice versa.) For example, the system

x + 2y − 3z = 4
5x − 6y + 8z = 9

is equivalent to $\begin{bmatrix} 1 & 2 & -3 \\ 5 & -6 & 8 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 4 \\ 9 \end{bmatrix}$

Observe that the system is completely determined by the matrix

$$M = [A, B] = \begin{bmatrix} 1 & 2 & -3 & 4 \\ 5 & -6 & 8 & 9 \end{bmatrix}$$

which is called the augmented matrix of the system.

10.8 TRANSPOSE

The transpose of a matrix A, written A^T, is the matrix obtained by writing the rows of A, in order, as columns. For example,

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \quad \text{and} \quad [1 \quad -3 \quad -5]^T = \begin{bmatrix} 1 \\ -3 \\ -5 \end{bmatrix}$$

Note that if A is an m×n matrix, then A^T is an n×m matrix. In particular, the transpose of a row vector is a column vector, and vice versa. Furthermore, if B = $[b_{ij}]$ is the transpose of A = $[a_{ij}]$, then $b_{ij} = a_{ji}$ for all i and j.

10.9 SQUARE MATRICES

A matrix with the same number of rows as columns is called a square matrix. A square matrix with n rows and n columns is said to be of order n, and is called an n-square matrix.

The main diagonal, or simply diagonal, of an n-square matrix A = $[a_{ij}]$ consists of the elements $a_{11}, a_{22}, \ldots, a_{nn}$, that is, the elements from the upper left corner to the lower right corner of the matrix. The trace of A, written tr(A), is the sum of the diagonal elements, that is, tr(A) = $a_{11} + a_{22} + \cdots + a_{nn}$.

The n-square unit matrix, denoted by I_n, or simply I, is the square matrix with 1's along the diagonal and 0's elsewhere. The unit matrix I plays the same role in matrix multiplication as the number 1 does in the usual multiplication of numbers. Specifically, for any matrix A,

$$AI = IA = A$$

Consider, for example, the matrices

$$\begin{bmatrix} 1 & -2 & 0 \\ 0 & -4 & -6 \\ 5 & 3 & 2 \end{bmatrix} \text{and} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Both are square matrices. The first is of order 3, and its diagonal consists of the elements 1, −4, 2 so its trace equals 1 − 4 + 2 = −1. The second matrix is of order 4; its diagonal consists only of 1's, and there are only 0's elsewhere. Thus the second matrix is the unit matrix of order 4.

10.9.1 Algebra of Square Matrices

Let A be any square matrix. Then we can multiply A by itself. In fact, we can form all nonnegative powers of A as follows:

$A^2 = AA, \qquad A^3 = A^2A, \ldots, \qquad A^{n+1} = A^nA, \ldots, \quad \text{and} \quad A^0 = I \text{ (when } A \neq 0)$

Polynomials in the matrix A are also defined. Specifically, for any polynomial

$f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n$

where the a_i are scalars, we define f (A) to be the matrix

$f(A) = a_0I + a_1A + a_2A^2 + \cdots + a_nA^n$

EXAMPLE 10.7 Suppose A = $\begin{bmatrix} 1 & 2 \\ 3 & -4 \end{bmatrix}$. Then

$$A^2 = \begin{bmatrix} 1 & 2 \\ 3 & -4 \end{bmatrix}\begin{bmatrix} 1 & 2 \\ 3 & -4 \end{bmatrix} = \begin{bmatrix} 7 & -6 \\ -9 & 22 \end{bmatrix} \quad \text{and}$$

$$A^3 = A^2A = \begin{bmatrix} 7 & -6 \\ -9 & 22 \end{bmatrix}\begin{bmatrix} 1 & 2 \\ 3 & -4 \end{bmatrix} = \begin{bmatrix} -11 & 38 \\ 57 & -106 \end{bmatrix}$$

Suppose f (x) = $2x^2 - 3x + 5$. Then

$$f(A) = 2\begin{bmatrix} 7 & -6 \\ -9 & -22 \end{bmatrix} - 3\begin{bmatrix} 1 & 2 \\ 3 & -4 \end{bmatrix} + 5\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 16 & -18 \\ -27 & 61 \end{bmatrix}$$

Suppose g(x) = $x^2 + 3x - 10$. Then

$$g(A) = \begin{bmatrix} 7 & -6 \\ -9 & -22 \end{bmatrix} + 3\begin{bmatrix} 1 & 2 \\ 3 & -4 \end{bmatrix} - 10\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Thus A is a zero of the polynomial g(x).

10.10 INVERTIBLE (NONSINGULAR) MATRICES, INVERSES

A square matrix A is said to be invertible (or nonsingular) if there exists a matrix B such that

AB = BA = I, (the identity matrix).

Such a matrix B is unique; it is called the inverse of A and is denoted by A^{-1}. Observe that B is the inverse of A if and only if A is the inverse of B. For example, suppose

$$A = \begin{bmatrix} 2 & 5 \\ 1 & 3 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 3 & -5 \\ -1 & 2 \end{bmatrix}$$

Then

$$AB = \begin{bmatrix} 6-5 & -10+10 \\ 3-3 & -5+6 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \text{and } BA = \begin{bmatrix} 6-5 & 15-15 \\ -2+2 & -5+6 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Thus A and B are inverses.

It is known that AB = I if and only if BA = I ; hence it is only necessary to test one product to determine whether two matrices are inverses. For example,

$$\begin{bmatrix}1 & 0 & 2\\ 2 & -1 & 3\\ 4 & 1 & 8\end{bmatrix}\begin{bmatrix}-11 & 2 & 2\\ -4 & 0 & 1\\ 6 & -1 & -1\end{bmatrix} = \begin{bmatrix}-11+0+12 & 2+0-2 & 2+0-2\\ -22+4+18 & 4+0-3 & 4-1-3\\ -44-4+48 & 8+0-8 & 8+1-8\end{bmatrix} = \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}$$

Thus the two matrices are invertible and are inverses of each other.

10.11 DETERMINANTS

To each n-square matrix A = [a$_{ij}$] we assign a specific number called the determinant of A and denoted by det(A) or |A| or

$$\begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1n}\\ a_{21} & a_{22} & \cdots & a_{2n}\\ \multicolumn{4}{c}{\dotfill}\\ a_{n1} & a_{n2} & \cdots & a_{nn}\end{vmatrix}$$

We emphasize that a square array of numbers enclosed by straight lines, called a determinant of order n, is not a matrix but denotes the number that the determinant function assigns to the enclosed array of numbers, i.e., the enclosed square matrix.

The determinants of order 1, 2, and 3 are defined as follows:

$$|a_{11}| = a_{11} \qquad \begin{vmatrix} a_{11} & a_{12}\\ a_{21} & a_{22}\end{vmatrix} = a_{11}a_{22} - a_{12}a_{21}$$

$$\begin{vmatrix} a_{11} & a_{12} & a_{13}\\ a_{21} & a_{22} & a_{23}\\ a_{31} & a_{32} & a_{33}\end{vmatrix} = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31} - a_{12}a_{21}a_{33} - a_{11}a_{23}a_{32}$$

The diagram in Fig. A-3(a) may help the reader remember the determinant of order 2. That is, the determinant equals the product of the elements along the plus-labeled arrow minus the product of the elements along the minus-labeled arrow. There is an analogous diagram to remember a determinant of order 3 which appears in Fig. A-3(b).

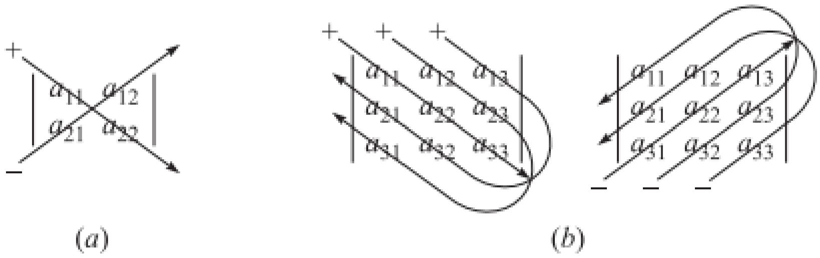

+
a_{11} a_{12}
a_{21} a_{22}
−
(a)
+ + +
a_{11} a_{12} a_{13}
a_{21} a_{22} a_{23}
a_{31} a_{32} a_{33}
− − −
(b)

Fig. A-3

EXAMPLE 10.8

(a) $\begin{vmatrix}5 & 4\\ 2 & 3\end{vmatrix} = 5(3) - 4(2) = 15 - 8 = 7,\quad \begin{vmatrix}2 & 1\\ -4 & 6\end{vmatrix} = 2(6) - 1(-4) = 12 + 4 = 16.$

(b) $\begin{vmatrix}2 & 1 & 3\\ 4 & 6 & -1\\ 5 & 1 & 0\end{vmatrix} = 2(6)(0) + 1(-1)(5) + 3(1)(4) - 5(6)(3) - 1(-1)(2) - 0(1)(4)$

$$= 0 - 5 + 12 - 90 + 2 - 0 = -81$$

Theorem 10.3: Let A and B be any n-square matrices. Then

det(AB) = det(A) · det(B)

10.11.1 Determinants and Inverses of 2 × 2 Matrices

Consider an arbitrary 2 × 2 matrix A = $\begin{bmatrix}a & b\\ c & d\end{bmatrix}$. Suppose |A| = ad − bc ≠ 0. Then one can prove that

$$A^{-1} = \begin{bmatrix}a & b\\ c & d\end{bmatrix}^{-1} = \begin{bmatrix}d/|A| & -b/|A|\\ -c/|A| & a/|A|\end{bmatrix} = \frac{1}{|A|}\begin{bmatrix}d & -b\\ -c & a\end{bmatrix}$$

In other words, when |A| ≠ 0, the inverse of a 2 × 2 matrix A is obtained as follows:

(1) Interchange the elements on the main diagonal.
(2) Take the negatives of the other elements.
(3) Multiply the matrix by 1/|A| or, equivalently, divide each element by |A|.

For example, if A = $\begin{bmatrix}2 & 3\\ 4 & 5\end{bmatrix}$, then $|A| = -2$ and so

$$A^{-1} = \frac{1}{-2}\begin{bmatrix}5 & -3\\ -4 & 2\end{bmatrix} = \begin{bmatrix}-\frac{5}{2} & \frac{3}{2}\\ 2 & -1\end{bmatrix}$$

On the other hand, if |A| = 0, then A^{-1} does not exist.

Theorem 10.4: A matrix A is invertible if and only if it has a nonzero determinant.

10.12 BOOLEAN (ZERO-ONE) MATRICES

The binary digits or bits are the symbols 0 and 1. Consider the following operations on these digits:

+	0	1
0	0	1
1	1	1

×	0	1
0	0	0
1	0	1

Viewing these bits as logical values (0 representing FALSE and 1 representing TRUE), the above operations correspond, respectively, to the logical operations of OR (∨) and AND (∧); that is,

∨	F	T
F	F	T
T	T	T

∧	F	T
F	F	F
T	F	T

The above operations on 0 and 1 are called Boolean operations since they also correspond to the operations of a Boolean algebra.

Now let A = $\left[a_{ij}\right]$ be a matrix whose entries are the bits 0 and 1 subject to the above Boolean operations. Then A is called a Boolean matrix. The Boolean product of two such matrices is the usual product except that now we use the Boolean operations of addition and multiplication. For example, if

$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$ and B=$\begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$, then $AB = \begin{bmatrix} 0+0 & 1+1 \\ 0+0 & 1+0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$

One can easily show that if A and B are Boolean matrices, then the Boolean product AB can be obtained by finding the usual product of A and B and then replacing any nonzero digit by 1.