Discrete Mathematics

Dr. rer. nat. Faten Abu-Shoga

August 29, 2023

Contents

Chapter 1

The Foundations: Logic and Proofs

1.1 Propositional Logic

Definition 1.1.1. A proposition is a declarative sentence (a sentence that declares a fact) that is either true or false.

Remark 1.1.2. A proposition is also called a statement.

Example 1.1.3. The following sentences are propositions

(a) Gaza is a Palestinian city.

(b) $2 - 1$ equals 3.

(c) The equation $x^2 + 1 = 0$ has two real solutions.

(d) IUG is a Palestinian university.

(e) Earth is the closest planet to the sun.

Example 1.1.4. The following sentences are NOT propositions

(a) How are you?

(b) Gaza is a beautiful city.

(c) The sky is reach.

(d) 4+1.

(e) I will come to school next week.

(f) Would you visit us tomorrow.

(g) He lives in Gaza.

(h) $x^2 = 9$.

Remark 1.1.5. Commends, questions, and opinions are not propositions.

The area of logic that deals with propositions is called the propositional calculus or propositional logic. It was first developed systematically by the Greek philosopher Aristotle more than 2300 years ago.

1.1.1 Compound Propositions

We now turn our attention to methods for producing new propositions from those that we already have. These methods were discussed by the English mathematician George Boole in 1854 in his book (The Laws of Thought). Many mathematical statements are constructed by combining one or more propositions. New propositions, called compound propositions, are formed from existing propositions using logical operators.

A compound proposition is a proposition that has at least one logical operator (connective).

Example 1.1.6. The following sentences are compound statements

(a) Gaza is a Palestinian city and Palestine is an arabic country.

(b) $2 - 1$ equals 3 or 7 is divisible by 2.

(c) If 5 is an integer, then 5 is a real number.

(d) 2 divides 6 if and only if $2 \times 3 = 6$.

(e) π is not a rational number.

Notation 1.1.7. We will denote propositions by lowercase letters $p, q, r, ...$

1.1.2 Fundamental logical operators (connectives)

To form new compound statements out of old ones we use the following five fundamental logical operators:

1. (not) symbolized by $\neg$.
2. (and) symbolized by $\wedge$.
3. (or) symbolized by $\vee$.
4. (If..., then ...) symbolized by $\longrightarrow$.
5. (...if and only if ...) symbolized by $\longleftrightarrow$.

1.1.3 Truth tables

A truth table is a mathematical table used in logic which determines the truth values of a compound proposition form for all logical possibilities of its components. A truth table has one column for each component and one final column for the compound proposition. Each row of the truth table contains one possible truth value for each component and the result of the logical operation for those values. We will use (T) for true and (F) for false.

1.1.4 Negation

Definition 1.1.8. (Negation)
The connective $\neg$ is called the negation and it may be placed before any proposition p to form a compound proposition $\neg p$ (read: not p or the negation of p). The truth values for $\neg p$ are defined as follows:

p	$\neg p$
T	F
F	T

Remark 1.1.9. If p is a proposition, then $\neg p$ is the statement (It is not the case that p).

Example 1.1.10. Find the negation of each of the following propositions and express it in simple English:

(a) $\sqrt{2}$ is a rational number.

(b) Ahmed's PC runs Linux.

Proof. **(a)** The negation is (It is not the case that $\sqrt{2}$ is a rational number). More simply, ($\sqrt{2}$ is not a rational number).

(b) The negation is (It is not the case that Ahmed's PC runs Linux.) Or (Ahmed's PC does not run Linux.)

□

1.1.5 Conjunction

Definition 1.1.11. (Conjunction)
The connective $\wedge$ is called the conjunction and it may be placed between any two propositions p and q to form a compound proposition $p \wedge q$ (read: p and q or the conjunction of p and q). The truth values for $p \wedge q$ are defined as follows:

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

Remarks 1.1.12.

(1) The proposition $p \wedge q$ is true only when both p and q are true.

(2) In a compound proposition with two components p and q there are $2 \times 2 = 4$ possibilities, called the logical possibilities. In general, if a compound proposition has n components, then there are 2^n logical possibilities.

Example 1.1.13. Indicate which of the following proposition is T and which is F:

(a) $1 + 1 = 2$ and $3 - 1 = 2$.

(b) 5 is an integer and $1 - 3 = 1$.

(c) $5 - 0 = 4$ and $5 - 1 = 4$.

(d) $5 \times 2 = 5$ and $5 \times 3 = 10$.

Proof. □

Example 1.1.14. Construct a truth table for the compound proposition $p \wedge (\neg q)$.

Proof. □

Remark 1.1.15. The English words but, while, and although are usually translated symbolically with the conjunction connective, because they have the same meaning as and.

Example 1.1.16. Translate the following statement into logical form using connectives: "8 is divisible by 2 but it is not divisible by 3."

Proof. Let p symbolizes "8 is divisible by 2" and let q symbolizes "8 is divisible by 3." Then the statement may symbolized by $p \wedge (\neg q)$. □

1.1.6 Disjunction

In English language there is an ambiguity involved in the use of "or".

Inclusive or: The statement "I will get a Master degree or a Ph. D" indicate that the speaker will get both the Master degree and the Ph. D.

Exclusive or: But in the statement "I will study mathematics or physics " means that only one of the two fields will be chosen. In mathematics and logic we can not allow ambiguity. Hence we must decide on the meaning of the word "or".

Definition 1.1.17. (Disjunction)
The connective $\vee$ is called the disjunction and it may be placed between any two propositions p and q to form the compound proposition $p \vee q$ (read: p or q or the disjunction of p and q). The truth values for $p \vee q$ are defined as follows:

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

Remark 1.1.18. The proposition $p \vee q$ is true when at least one of p and q is true.

Example 1.1.19. Indicate which of the following propositions is T and which is F:

(a) $3 + 3 = 6$ or $1 - 1 = 1$.

(b) $4 > 4$ or $4 = 4$.

(c) $\sqrt{-1} = 2$ or $(2)^2 = -1$.

(d) 7 is a prime number or 7 is an odd number.

Proof. □

Example 1.1.20. Construct the truth table for the compound proposition $\neg[p \vee (\neg q)]$.

Proof. □

Definition 1.1.21. (Exclusive or)
The connective $\oplus$ is called the exclusive or and it may be placed between any two propositions p and q to form the compound proposition $p \oplus q$ (read: the exclusive or of p and q). The truth values for $p \oplus q$ are defined as follows:

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

1.1.7 Conditional Statements

We will discuss two other important ways to combine propositions.

Definition 1.1.22. (Conditional)
The connective $\rightarrow$ is called the conditional and it may be placed between any two propositions p and q to form the compound proposition $p \rightarrow q$ (read: if p then q). The truth values of $p \rightarrow q$ are defined by the following table:

p	q	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

Remarks 1.1.23.

(1) The proposition $p \rightarrow q$ is false only when p is true and q is false.

(2) In a conditional proposition $p \rightarrow q$, p is called the hypothesis (or antecedent or premise) and q is called the conclusion (or consequent).

(3) A conditional statement $p \to q$ is also called an implication.

Example 1.1.24. Determine whether the following propositions are T or F:

(a) If $2 - 4 = 2$, then $2 - 2 = 4$.

(b) If $7 < 9$, then $7 < 8$.

(c) If $3 > 3$, then $4 > 3$.

(d) If $5 < 6$, then 5 is even.

Proof. □

Example 1.1.25. Construct the truth table for the compound proposition $(p \vee q) \to r$.

Proof. □

Example 1.1.26. Let p be the statement (Ali learns discrete mathematics) and q the statement (Ali will find a good job.) Express the statement $p \longrightarrow q$ as a statement in English.

Proof. The statement $p \longrightarrow q$ represents the statement

(If Ali learns discrete mathematics, then he will find a good job.)

There are many other ways to express this conditional statement in English. Among the most natural of these are:

(Ali will find a good job when he learns discrete mathematics.)

(For Ali to get a good job, it is sufficient for him to learn discrete mathematics.)

(Ali will find a good job unless he does not learn discrete mathematics.) □

Remark 1.1.27. We use $p \to q$ to translate the following propositions:

1. If p, then q.
2. p only if q.
3. p implies q.
4. p is sufficient to q.
5. q if p.
6. q is necessary for p.
7. q whenever p.
8. q when p.
9. q unless $\neg p$.

Remark 1.1.28. The if-then construction used in many programming languages is different from that used in logic. Most programming languages contain statements such as if p then S , where p is a proposition and S is a program segment (one or more statements to be executed). When execution of a program encounters such a statement, S is executed if p is true, but S is not executed if p is false.

Example 1.1.29. What is the value of the variable x after the statement (if x -4 = 1 then $x := x + 1$) if $x = 5$ before this statement is encountered? (The symbol := stands for assignment. The statement $x := x + 1$ means the assignment of the value of $x + 1$ to x.)

Proof. Since $5 - 1 = 4$ is true, the assignment statement $x := x + 1$ is executed. Hence, x has the value 6 after this statement is encountered. □

1.1.8 Converse, Contrapositive, and Inverse

We can form some new conditional statements starting with a conditional statement $p \longrightarrow q$. In particular, there are three related conditional statements that occur so often that they have special names.

Definition 1.1.30. (Converse, contrapositive, and inverse of a conditional statement)
Let p and q be propositions.

(1) The converse of $p \to q$ is $q \to p$.

(2) The contrapositive of $p \to q$ is $\neg q \to \neg p$.

(3) The inverse of $p \to q$ is $\neg p \to \neg q$.

Example 1.1.31. Find the contrapositive, the converse, and the inverse of the conditional statement (The home team wins whenever it is raining.)

Proof. Since (q whenever p) is one of the ways to express the conditional statement $p \longrightarrow q$, the original statement can be rewritten as (If it is raining, then the home team wins.)

Consequently, the contrapositive of this conditional statement is (If the home team does not win, then it is not raining.)

The converse is (If the home team wins, then it is raining.) The inverse is (If it is not raining, then the home team does not win.)

Only the contrapositive always has the same truth value as the original statement. □

1.1.9 Biconditionals

We now introduce a new way to combine propositions that expresses that two propositions have the same truth value.

Definition 1.1.32. (Biconditional)
The connective $\leftrightarrow$ is called the biconditional and it may be placed between any two propositions p and q to form the compound proposition $p \leftrightarrow q$ (read: p if and only if q). The truth values of $p \to q$ are given by the following table:

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

Remarks 1.1.33.

(1) The proposition $p \leftrightarrow q$ is true when both p and q have the same truth values.

(2) The proposition $p \leftrightarrow q$ also called bi-implication.

Example 1.1.34. Determine whether the following propositions are T or F:

(a) 1 is odd if and only if 3 is even.

(b) $|5| = -5$ if and only if $5 > 0$.

(c) $\sqrt{4} = 2$ if and only if $(2)^2 = 4$.

(d) $5 > 6$ if and only if 5 is even.

Proof. □

Example 1.1.35. Construct the truth table for the compound proposition $(p \wedge q) \leftrightarrow p$.

Proof. □

Remark 1.1.36. We use $p \leftrightarrow q$ to translate the following propositions:

1. p if and only if q.
2. p is equivalent to q.
3. p is necessary and sufficient for q.

Example 1.1.37. Translate the given compound propositions into a symbolic form using the given symbols.

(a) (A natural number is even if and only if it is divisible by 2.) (E, D)

(b) (A matrix has an inverse whenever its determinant is not zero.) (I, Z)

(c) (A function is differentiable at a point only if it is continuous at that point.) (D, C)

Proof. **(a)** Let p denotes (A natural number is even) and let q denotes (A natural number is divisible by 2.) Then the proposition can be written as $E \leftrightarrow D$.

(b) Let q denotes (A matrix has an inverse) and let p denotes (The determinant of a matrix is not zero.) Then a symbolic form of the proposition is $p \rightarrow q$.

(c) Let p be (A function is differentiable at a point) and let q be (A function is continuous at that point.) Then a symbolic form of the proposition is $p \rightarrow q$.

□

1.1.10 Implicit Use of Biconditionals

You should be aware that biconditionals are not always explicit in natural language. In particular, the "if and only if" construction used in biconditionals is rarely used in common language. Instead, biconditionals are often expressed using an "if, then" or an "only if" construction. The other part of the "if and only if" is implicit. That is, the converse is implied, but not stated. For example, consider the statement in English "If you finish your meal, then you can have dessert." What is really meant is "You can have dessert if and only if you finish your meal." This last statement is logically equivalent to the two statements "If you finish your meal, then you can have dessert" and "You can have dessert only if you finish your meal." Because of this imprecision in natural language, we need to make an assumption whether a conditional statement in natural language implicitly includes its converse. Because precision is essential in mathematics and in logic, we will always distinguish between the conditional statement $p \longrightarrow q$ and the biconditional statement $p \longleftrightarrow q$.

1.1.11 Precedence of Logical Operators

We can construct compound propositions using the negation operator and the logical operators defined so far. We will generally use parentheses to specify the order in which logical operators in a compound proposition are to be applied. However, to reduce the number of parentheses, we specify that the negation operator is applied before all other logical operators. Another general rule of precedence is that the conjunction operator $\wedge$ takes precedence over the disjunction operator $\vee$, so that $p \wedge q \vee r$ means $(p \wedge q) \vee r$ rather than $p \wedge (q \vee r)$. Because this rule may be difficult to remember, we will continue to use parentheses so that the order of the disjunction and conjunction operators is clear. Finally, it is an accepted rule that the conditional and biconditional operators $\longrightarrow$ and $\longleftrightarrow$ have lower precedence than the conjunction and disjunction operators, $\wedge$ and $\vee$. Consequently, $p \vee q \longrightarrow r$ is the same as $(p \vee q) \longrightarrow r$. We will use parentheses when the order of the conditional operator and biconditional operator is at issue, although the conditional operator has precedence over the biconditional operator.

1.1.12 Logic and Bit Operations

Computers represent information using bits. A bit is a symbol with two possible values, namely, 0 (zero) and 1 (one). This meaning of the word bit comes from binary digit, because zeros and ones are the digits used in binary representations of numbers. The well-known statistician John Tukey introduced this terminology in 1946. A bit can be used to represent a truth value, because there are two truth values, namely, true and false. As is customarily done, we will us e a 1 bit to represent true an d a 0 bit to represent false. That is, 1 represents T (true), 0 represents F (false). A variable is called a Boolean variable if its value is either true or false. Consequently, a Boolean variable can be represented using a bit.

Computer bit operations correspond to the logical connectives. By replacing true by a one and false by a zero in the truth tables for the operators $\wedge$, $\vee$, and $\oplus$, we obtain the following tables

x	y	$x \vee y$	$x \wedge y$	$x \oplus y$
1	1	1	1	0
1	0	1	0	1
0	1	1	0	1
0	0	0	0	0

We will also use the notation OR , AND , and XOR for the operators $\wedge$, $\vee$, and $\oplus$, as is done in various programming languages.

Definition 1.1.38. (Bit string)
A bit string is a sequence of zero or more bits. The length of this string is the number of bits in the string.

Example 1.1.39. 0010111 is a bit string of length 7.

We can extend bit operations to bit strings.

Definition 1.1.40. (Bitwise operators)
We define the bitwise OR , bitwise AND , and bitwise XOR of two strings of the same length to be the strings that have as their bits the OR , AND , and XOR of the corresponding bits in the two strings, respectively.

Notation 1.1.41. We use the symbols $\wedge$, $\vee$, and $\oplus$ to denote the bitwise OR , bitwise AND , and bitwise XOR operations, respectively.

Remark 1.1.42. Throughout this course, we will split bit strings into blocks of four bits to make them easier to read.

Example 1.1.43. Find the bitwise OR , bitwise AND , and bitwise XOR of the bit strings $x = 01\ 1011\ 0110$ and $y = 11\ 0001\ 1101$.

Proof.

x	01 1011 0110
y	11 0001 1101
$x \vee y$	11 1011 1111
$x \wedge y$	01 0001 0100
$x \oplus y$	10 1010 1011

□

Example 1.1.44. Evaluate the expression $1101 \wedge (0101 \vee 1001)$.

Proof.

x	y	z	$y \vee z$	$x \wedge (y \vee z)$
1001	0101	1000	1101	1001

□

Exercises

Page 12: 1-35, 42-44

1.2 Applications of Propositional Logic

Translating English Sentences

System Specifications

Boolean Searches

Logic Circuits

1.3 Propositional Equivalences

Definition 1.3.1.

(2) The symbol $\equiv$ is not a logical connective, and $p \equiv q$ is not a compound proposition.

(3) The symbol $\Longleftrightarrow$ is sometimes used instead of $\equiv$ to denote logical equivalence.

Example 1.3.6. Show that $\sim [p \vee (\sim q)] \equiv \sim p \wedge q$.

Proof. □

Example 1.3.7. Determine whether $p \to q$ and its converse are logically equivalent or not.

Proof. □

Theorem 1.3.8. *Let p and q be any two statements and let T denote the proposition that is always true and F denote the proposition that is always false. Then*

1	*Identity laws:*	$p \wedge T \equiv p$ $p \vee F \equiv p$
2	*Domination laws:*	$p \wedge F \equiv F$ $p \vee T \equiv T$
3	*Laws of idempotency:*	$(p \vee p) \equiv p$ $p \wedge p \equiv p$
4	*Law of double negation:*	$\neg(\neg p) \equiv p$
5	*Commutative laws:*	$(p \wedge q) \equiv (q \wedge p)$ $(p \vee q) \equiv (q \vee p)$
6	*Associative laws:*	$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$ $(p \vee q) \vee r \equiv p \vee (q \vee r)$
7	*Distributive laws:*	$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$ $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
8	*De Morgan's laws:*	$\neg(p \wedge q) \equiv \neg p \vee \neg q$ $\neg(p \vee q) \equiv \neg p \wedge \neg q$
9	*Absorption laws:*	$p \vee (p \wedge q) \equiv p$ $p \wedge (p \vee q) \equiv p$
10	*Negation laws:*	$p \wedge \neg p \equiv F$ $p \vee \neg p \equiv T$

Proof. By truth tables. □

1.3.2 Logical Equivalences Involving Conditional Statements

Theorem 1.3.9. *Let p, q, and r be any statements. Then*

Contrapositive law:	$p \to q \equiv \neg q \to \neg p$
	$p \to q \equiv \neg p \vee q$
	$[(p \to q) \wedge (p \to r)] \equiv [p \to (q \wedge r)]$
	$[(p \to r) \wedge (q \to r)] \equiv [(p \vee q) \to r]$
	$[(p \to q) \vee (p \to r)] \equiv [p \to (q \vee r)]$
	$[(p \to r) \vee (q \to r)] \equiv [(p \wedge q) \to r]$

Proof. By truth tables. □

1.3.3 Logical Equivalences Involving Biconditional Statements

Theorem 1.3.10. *Let p and q be any two statements. Then*

$p \leftrightarrow q \equiv (p \to q) \wedge (q \to p)$
$p \leftrightarrow q \equiv (p \wedge q) \vee (\neg p \wedge \neg q)$
$p \leftrightarrow q \equiv \neg p \leftrightarrow \neg q$

Proof. By truth tables. □

1.3.4 Using De Morgan's Laws

The two logical equivalences known as De Morgan's laws are particularly important. They tell us how to negate conjunctions and how to negate disjunctions. In particular, the equivalence $\neg(p \wedge q) \equiv \neg p \vee \neg q$ tells us that the negation of a disjunction is formed by taking the conjunction of the negations of the component propositions. Similarly, the equivalence $\neg(p \vee q) \equiv \neg p \wedge \neg q$ tells us that the negation of a conjunction is formed by taking the disjunction of the negations of the component propositions.

Example 1.3.11. Use De Morgan's laws to express the negations of "Ahmed is an IT student and he has a laptop computer".

Proof. Let p denote "Ahmed is an IT student" and q denote "Ahmed has a laptop computer." Then the sentence can be represented by $p \wedge q$. Using De Morgan's laws, the negation of the sentence is $\neg(p \wedge q) \equiv \neg p \vee \neg q$. Thus, we can express the negation of our statement as "Ahmed is not an IT student or he does not have a laptop computer" □

1.3.5 Constructing New Logical Equivalences

The laws summarized in Theorem 1, Theorem 2, and Theorem 3 are very useful tools for constructing logical equivalences. It should be noted that these rules are selected just for convenient references and are not intended to be independent of each other. We could use a truth table to show the logical equivalence of two propositions, but in the case of propositions with a large number of variables it is not practical.

We use the fact that a proposition in a compound proposition can be replaced by a compound proposition that is logically equivalent to it without changing the truth value of the original compound proposition. We also use the fact that if $p \equiv q$ and $q \equiv r$, then $p \equiv r$.

Example 1.3.12. Show that $\neg(p \to q) \equiv (p \wedge \neg q)$.

Proof. We will establish this equivalence by developing a series of logical equivalences, using one of the equivalences in Theorem 1, Theorem 2, and Theorem 3 at a time, starting with $\neg(p \to q)$ and ending with $p \wedge \neg q$.

$$\begin{array}{lcll} \neg(p \to q) & \equiv & \neg(\neg p \vee q) & \text{(by Theorem 2)} \\ & \equiv & \neg\neg p \wedge \neg q & \text{(by De Morgan law)} \\ & \equiv & p \wedge \neg q & \text{(by the double negation law)} \end{array}$$

□

Example 1.3.13. Show that $(p \wedge q) \to (p \vee q)$ is a tautology.

Proof. To show that this statement is a tautology, we will use logical equivalences to show that it is logically equivalent to T.

$$\begin{array}{lcll} (p \wedge q) \to (p \vee q) & \equiv & \neg(p \wedge q) \vee (p \vee q) & \text{(Theorem 2)} \\ & \equiv & (\neg p \vee \neg q) \vee (p \vee q) & \text{(De Morgan law)} \\ & \equiv & (\neg p \vee p) \vee (\neg q \vee q) & \text{(associative and commutative laws)} \\ & \equiv & T \vee T & \text{(Negation law)} \\ & \equiv & T & \text{(idempotency law)} \end{array}$$

□

Example 1.3.14. Show that

$$[(p \wedge q) \to r] \equiv [p \to (q \to r)]$$

by developing a series of logical equivalences.

Proof.

$$\begin{array}{lcll} (p \wedge q) \to r & \equiv & \neg(p \wedge q) \vee r & \text{(Theorem 2)} \\ & \equiv & (\neg p \vee \neg q) \vee r & \text{(De Morgan law)} \\ & \equiv & \neg p \vee (\neg q \vee r) & \text{(associative law)} \\ & \equiv & \neg p \vee (q \ \to r) & \text{(Theorem 2)} \\ & \equiv & p \to (q \to r) & \text{(Theorem 2)} \end{array}$$

□

Exercises

Page 34: $1 - 33$

1.4 Predicates and Quantifiers

In this section we will introduce a more powerful type of logic called predicate logic. We will see how predicate logic can be used to express the meaning of a wide range of statements in mathematics and computer science in ways that permit us to reason and explore relationships between objects. To understand predicate logic, we first need to introduce the concept of a predicate. Afterward, we will introduce the notion of quantifiers, which enable us to reason with statements that assert that a certain property holds for all objects of a certain type and with statements that assert the existence of an object with a particular property.

1.4.1 Predicates

Some sentences depend on some variables and become propositions when the variables are replaced by a certain values.

Definition 1.4.1. (Propositional function)
A sentence containing one or more variables and which becomes a proposition only when the variables are replaced by certain values is called a propositional function (predicate) (or an open sentence).

Notation 1.4.2. A propositional function P with variables $x_1, x_2, \cdots, x_n$ will be denoted by $P(x_1, x_2, \cdots, x_n)$.

Example 1.4.3. $P(x) : x + 1 = 0$ is a propositional function.
$P(0)$ is false but $P(-1)$ is true.

Example 1.4.4. $P(x, y) : x + y = 1$ is a propositional function.
$P(0, 1)$ is true but $P(1, 1)$ is false.

Example 1.4.5. Consider the statement "if $x > 0$ then $x := x + 1$." When this statement is encountered in a program, the value of the variable x at that point in the execution of the program is inserted into $P(x) : x > 0$. If $P(x)$ is true for this value of x, the assignment statement $x := x + 1$ is executed, so the value of x is increased by 1. If $P(x)$ is false for this value of x, the assignment statement is not executed, so the value of x is not changed.

Definition 1.4.6. (Universe)
The set of all objects that can be considered in a propositional function is called the universe or the domain of discourse.

Remark 1.4.7. In many cases the universe will be understood from the context. However, there are times when it must be specified explicitly.

1.4.2 Universal and existence quantifiers

When the variables in a propositional function are assigned values, the resulting statement becomes a proposition with a certain truth value. However, there is another important way, called quantification, to create a proposition from a propositional function. Quantification expresses the extent to which a predicate is true over a range of elements. In English, the words all, some, many, none, and few are used in quantifications. We will focus on two types of quantification here: universal quantification, which tells us that a predicate is true for every element under consideration, and existential quantification, which tells us that there is one or more element under consideration for which the predicate is true. The area of logic that deals with predicates and quantifiers is called the predicate calculus.

Definition 1.4.8. (Universal and existence quantifiers)
Let $P(x)$ be a propositional function.

(1) $\forall$ is called the universal quantifier and the proposition $(\forall x)(P(x))$ (read: for all x $P(x)$) is true when $P(x)$ is true for all x in the universe.

(2) $\exists$ is called the existential quantifier and the statement $(\exists x)(P(x))$ (read: there exists x such that $P(x)$) is true when there exists at least one x in the universe such that $P(x)$ is true.

Statement	When True?	When False?
$(\forall x)(P(x))$	$P(x)$ is true for all x in the universe	There is an x for which $P(x)$ is false
$(\exists x)(P(x))$	There is an x such that $P(x)$ is true	$P(x)$ is false for all x in the universe

Example 1.4.9. Determine whether the following statements are true or false:

(a) Let $U = \{1, 2, 3, 4\}$ be the domain of discourse.

- $(\forall x)(x + 2 \in U)$
- $(\exists x)(x + 1 = 4)$

(b) Let $U = \mathbb{R}$ be the universe.

- $(\exists x)(x \geq -1)$

- $(\forall x)(x \geq 5)$
- $(\forall x)(|x| \geq 0)$
- $(\exists x)(|x| < 0)$
- $(\forall x)(x > 0 \text{ or } x < 0)$.

Remarks 1.4.10.

(1) "for every", "for each", "for arbitrary", "for any", and "for all" have the same meaning in mathematics.

(2) "for some", "there is", "at least one" and "there exists" have the same meaning in mathematics.

(3) In less formal expressions, we often put the quantifier after the sentence.

Example 1.4.11. $[(\forall x)(f(x) = 0)] \equiv [f(x) = 0 \ \ \forall x]$
$(\exists x)(f(x) = 0) \equiv [f(x) = 0 \text{ for some } x]$

Remarks 1.4.12. Let the domain of discourse be $U = \{a_1, a_2, \cdots, a_n\}$. Then

(1) The statement $(\forall x)(P(x))$ means $P(a_1) \wedge P(a_2) \wedge \cdots \wedge P(a_n)$.

(2) The statement $(\exists x)(P(x))$ means $P(a_1) \vee P(a_2) \vee \cdots \vee P(a_n)$.

Example 1.4.13. Let the domain of discourse be $U = \{0, 1, 2\}$ and $P(x)$ be the statement $x < 3$.

(a) The statement $(\forall x)P(x)$ is the same as the conjunction $P(0) \wedge P(1) \wedge P(2)$ and it is true.

(b) The statement $(\exists x)P(x)$ is the same as the disjunction $P(0) \vee P(1) \vee P(2)$ and it is true.

1.4.3 Quantifiers with Restricted Domains

An abbreviated notation is often used to restrict the domain of a quantifier. In this notation, a condition a variable must satisfy is included after the quantifier.

Example 1.4.14. Let the domain consist of the real numbers $\mathbb{R}$. The statement $(\forall x \geq 0)(\sqrt{x} \in \mathbb{R})$ states "The square root of a nonnegative real number is a real number." It has the same meaning as $(\forall x)(x \geq 0 \rightarrow \sqrt{x} \in \mathbb{R})$.

Example 1.4.15. Let the domain consist of the natural numbers $\mathbb{N}$. The statement $(\exists x < 5)(x^2 = 36)$ states "There is a natural number x with $x < 5$ such that $x^2 = 36$." It has the same meaning as $(\exists x)(x < 5 \wedge x^2 = 36)$.

Remark 1.4.16. Let U be the domain of discourse and let $A \subseteq U$. Statements of the form "Every element of the set A has the property P" and "Some element of the set A has property P" occur so frequently that abbreviated symbolic forms are desirable.

(a) "Every element of the set A has the property P" could be restated as "If $x \in A$, then . . ." and symbolized by

$$(\forall x \in A)(P(x)) \quad \text{or} \quad (\forall x)(x \in A \to P(x)).$$

(b) "Some element of the set A has property P" is abbreviated by

$$(\exists x \in A)(P(x)) \quad \text{or} \quad (\exists x)(x \in A \wedge P(x)).$$

1.4.4 Precedence of Quantifiers

The quantifiers $\forall$ and $\exists$ have higher precedence than all logical operators from propositional calculus. For example, $(\forall x)P(x) \vee Q(x)$ is the disjunction of $(\forall x)P(x)$ and $Q(x)$. In other words, it means $(\forall xP(x)) \vee Q(x)$ rather than $(\forall x)(P(x) \vee Q(x))$.

1.4.5 Quantifier Negation

The rules for negations for quantifiers are called De Morgan's laws for quantifiers.

Definition 1.4.17. (De Morgan's Laws for Quantifiers)

(1) $\neg[(\forall x)(P(x))] \equiv (\exists x)(\neg P(x))$

(2) $\neg[(\exists x)(P(x))] \equiv (\forall x)(\neg P(x))$

Example 1.4.18. Which of the following is equivalent to the negation of the statement "All functions are continuous"

(a) All functions are not continuous.

(b) Some functions are continuous.

(c) Some functions are not continuous.

Proof. The universe is the collection of all functions.
$P(x)$: x is continuous.
Statement: $(\forall x)(P(x))$
Negation: $(\exists x)(\sim P(x))$
Thus (c) is true. □

Example 1.4.19. Find an equivalent statement to the negation of the statement "Some rational numbers are integers" by using quantifier negation.

Proof. The universe is $\mathbb{Q}$.
$P(x)$: x is an integer.
Statement: $(\exists x)(P(x))$
Negation: $(\forall x)(\sim P(x))$
Thus a denial is "All rational numbers are not integers". □

Example 1.4.20. State in words the negation of the statement "For all $x \in \mathbb{Z}$, if x is divisible by 6, then x is divisible by 3".

Proof. $P(x)$: x is divisible by 6.
$Q(x)$: x is divisible by 3.
Statement: $(\forall x \in \mathbb{Z})(P(x) \longrightarrow Q(x))$.
Negation: $(\exists x \in \mathbb{Z})(P(x) \wedge \sim Q(x))$.
Thus the negation is "There exist an integer that is divisible by 6 and is not divisible by 3". □

1.4.6 Using Quantifiers in System Specifications

In Section 1.2 we used propositions to represent system specifications. However, many system specifications involve predicates and quantifications.

Example 1.4.21. Use predicates and quantifiers to express the following system specifications:

(a) "Every mail message larger than one megabyte will be compressed."

(b) "If a user is active, at least one network link will be available."

Proof. (a) Let $S(m)$ be "Mail message m is larger than 1 megabytes," where the variable m has the domain of all mail messages. Let $C(m)$ denote "Mail message m will be compressed." Then the specification "Every mail message larger than one megabyte will be compressed" can be represented as $(\forall m)(S(m) \rightarrow C(m))$.

(b) Let $A(u)$ represent "User u is active," where the variable u has the domain of all users, let $S(n)$ denote "Network link n is available," where n has the domain of all network links. Then the specification "If a user is active, at least one network link will be available" can be represented by $(\exists u)A(u) \rightarrow (\exists n)S(n)$.

□

Exercises

Page 53: 1-30

1.5 Omit

1.6 Rules of Inference

One of the most important task of a logician is the testing of arguments. An argument is a sequence of statements that end with a conclusion. An argument is considered to be valid if final statement (the conclusion) follows from other statements, called the hypotheses or premises. The conjunction of the hypotheses implies the conclusion. That is, an argument is valid if and only if it is impossible for all the premises to be true and the conclusion to be false.

Example 1.6.1. The following is an argument in which the first two statements are hypotheses, and the last statement is the conclusion.

- All men are tall.
- Ali is a man.
- Therefore Ali is tall.

The above argument may be symbolized as follows:

- p1. All men are tall.
- p2. Ali is a man.

  ———————————————

- C $\quad \therefore$ Ali is tall.

Example 1.6.2. The following is an argument in which the first four statements are hypotheses, and the last statement is the conclusion.

- If he studies IT, then he will earn a good life.
- If he studies mathematics, then he will be happy.
- If he will earn a good life or he will be happy, then his university tuition is not wasted.
- His university tuition is wasted.
- Therefore, he studies neither IT nor mathematics.

The above argument may be symbolized as follows:
p1. $I \rightarrow E$.
p2. $M \rightarrow H$.
p3. $(E \vee H) \rightarrow \sim W$.
p4. W.
C $\quad \therefore \ \sim I \wedge \sim M$

Definition 1.6.3.

(a) An argument in propositional logic is a sequence of propositions. All but the final proposition in the argument are called premises and the final proposition is called the conclusion.

(b) An argument is valid if the truth of all its premises implies that the conclusion is true.

(c) An argument form in propositional logic is a sequence of compound propositions involving propositional variables.

(d) An argument form is valid no matter which particular propositions are substituted for the propositional variables in its premises, the conclusion is true if the premises are all true.

From the definition of a valid argument form we see that an argument is considered to be valid if the conjunction of the premises implies the conclusion. That is, if an argument form has premises $p_1, p_2, \cdots, p_n$ and conclusion q, then the argument is valid when

$$(p_1 \wedge p_2 \wedge \cdots \wedge p_n) \longrightarrow q$$

is a tautology.

The key to showing that an argument in propositional logic is valid is to show that its argument form is valid. Consequently, we would like techniques to show that argument forms are valid. We will now develop methods for accomplishing this task.

1.6.1 Rules of Inference for Propositional Logic

To establish the validity of an argument form by means of a truth table may be a tedious approach. For example, if an argument form involves n different propositional variables, then we need a truth table with 2^n rows. Fortunately we can prove that an argument form is valid by deducing the conclusion from the premises in a few steps using the rules of inference.

The modus ponens rule (or the law of detachment) is the most fundamental rule of reasoning. Modus ponens is Latin for "The way that affirms." It is based on the tautology

$$[p \wedge (p \to q)] \to q.$$

This tautology leads to the following valid argument form, which we have already seen in our initial discussion about arguments (where the symbol $\therefore$ denotes "therefore"):

$$\begin{array}{l} p \\ p \to q \\ \hline \therefore q \end{array}$$

Using this notation, the hypotheses are written in a column, followed by a horizontal bar, followed by a line that begins with the therefore symbol and ends with the conclusion. In particular, modus ponens tells us that if a conditional statement and the hypothesis of this conditional statement are both true, then the conclusion must also be true.

Example 1.6.4. Find the argument form for the following argument and determine whether it is valid. Can we conclude that the conclusion is true if the premises are true?

- If 8 is divisible by 2, then 8 is an even number.
- We know that 8 is divisible by 2.
- Therefore, 8 is an even number.

Proof. Let p denote "8 is divisible by 2" and let q denote "8 is an even number. Then the argument can be represented as

p1. $p \longrightarrow q$

p2. p

C. $\therefore\ q$

This argument form is modus ponens and it is valid. Since the two premises are true, the the conclusion is true. □

Rules of Inference

Name	Tautology	Rule of Inference
Modus Ponens	$[p \wedge (p \to q)] \to q$	$\begin{array}{l} p \\ \underline{p \to q} \\ \therefore q \end{array}$
Modus Tollens	$[\neg q \wedge (p \to q)] \to \neg p$	$\begin{array}{l} \neg q \\ \underline{p \to q} \\ \therefore \neg p \end{array}$
Transitive law (Hypothetical syllogism)	$[(p \to q) \wedge (q \to r)] \to (p \to r)$	$\begin{array}{l} p \to q \\ \underline{q \to r} \\ \therefore p \to r \end{array}$
Disjunctive syllogism	$[(p \vee q) \wedge \neg p] \to q$	$\begin{array}{l} p \vee q \\ \underline{\neg p} \\ \therefore q \end{array}$
Law of addition	$p \to (p \vee q)$	$\begin{array}{l} p \\ \hline \therefore p \vee q \end{array}$
Law of simplification	$(p \wedge q) \to p$	$\begin{array}{l} \underline{p \wedge q} \\ \therefore p \end{array}$
Law of conjunction	$(p \wedge q) \to (p \wedge q)$	$\begin{array}{l} p \\ \underline{q} \\ \therefore p \wedge q \end{array}$
Resolution	$[(p \vee q) \wedge (\neg p \vee r)] \to (q \vee r)$	$\begin{array}{l} p \vee q \\ \underline{\neg p \vee r} \\ \therefore q \vee r \end{array}$

Example 1.6.5. State which rule of inference is the basis of the following argument: "It is below freezing now. Therefore, it is either below freezing or raining now."

Proof. Let p be the proposition "It is below freezing now" and q the proposition "It is raining

now." Then this argument is of the form

$$\begin{array}{l} p \\ \hline \therefore p \vee q \end{array}$$

This is an argument that uses the addition rule. □

Example 1.6.6. State which rule of inference is the basis of the following argument: "If you study computer science, then you get a job. If you get a job, then you are successful. Therefore, if you study computer science, you are successful."

Proof. Let p be the proposition " you study computer science", q be the proposition "you get a job", and r be the proposition "you are successful." Then this argument is of the form

$$\begin{array}{l} p \longrightarrow q \\ q \longrightarrow r \\ \hline \therefore p \longrightarrow r \end{array}$$

This is an argument that uses the Transitive law. □

1.6.2 Using Rules of Inference to Build Arguments

When there are many premises, several rules of inference are often needed to show that an argument is valid. The following examples also show how arguments in English can be analyzed using rules of inference.

Example 1.6.7. Use rules of inference to show that the hypotheses imply the conclusion
Ahmed works hard
If Ahmed works hard, then he is a good boy
If Ahmed is a good boy, then he will get the job
Therefore Ahmed will get the job.

Proof. Let p denote "Ahmed works hard", q denote "Ahmed is a good boy", and r denote "Ahmed will get the job". Then the argument can be represented as

$$\begin{array}{l} p \\ p \longrightarrow q \\ q \longrightarrow r \\ \hline \therefore r \end{array}$$

The following argument form shows that the premises lead to the desired conclusion.

	Step	Reason
1.	p	Premise
2.	$p \longrightarrow q$	Premise
3.	$q \longrightarrow r$	Premise
4.	q	$1, 2,$ Modus Ponens
5.	r	$4, 3,$ Modus Ponens

□

Example 1.6.8. Use rules of inference to show that the following argument is a valid argument:

$$W \vee (H \wedge L)$$
$$(W \vee H) \rightarrow D / \therefore W \vee D.$$

Proof. 1. $W \vee (H \wedge L)$

2. $(W \vee H) \rightarrow D / \therefore W \vee D$

3. $(W \vee H) \wedge (W \vee L)$ 1, distributive law

4. $W \vee H$ 3, simplification law

5. D 2,4, Modus Ponens

6. $D \vee W$ 5, addition

7. $W \vee D$ 6, commutative law

□

1.6.3 Resolution

Computer programs have been developed to automate the task of reasoning and proving theorems. Many of these programs make use of a rule of inference known as resolution. This rule of inference is based on the tautology $[(p \vee q) \wedge (\neg p \vee r)] \rightarrow (q \vee r)$. The final disjunction in the resolution rule, $q \vee r$, is called the resolvent. When we let $q = r$ in this tautology, we obtain $[(p \vee q) \wedge (\neg p \vee q)] \rightarrow q$. Furthermore, when we let $r = F$, we obtain $[(p \vee q) \wedge \neg p] \rightarrow q$ (because $(q \vee F) \equiv q$, which is the tautology on which the rule of disjunctive syllogism is based.)

Example 1.6.9. Use resolution to show that the hypotheses "Jasmine is skiing or it is not snowing" and "It is snowing or Fatma is playing tennis" imply that "Jasmine is skiing or Fatma is playing tennis."

Proof. Let p be the proposition "It is snowing," q the proposition "Jasmine is skiing," and r the proposition "Fatma is playing tennis." We can represent the hypotheses as $\neg p \vee q$ and $p \vee r$, respectively. Using resolution, the proposition $q \vee r$, "Jasmine is skiing or Fatma is playing tennis," follows. □

Resolution plays an important role in programming languages based on the rules of logic, such as Prolog (where resolution rules for quantified statements are applied). Furthermore, it can be used to build automatic theorem proving systems. To construct proofs in propositional logic using resolution as the only rule of inference, the hypotheses and the conclusion must be expressed as clauses, where a clause is a disjunction of variables or negations of these variables. We can replace a statement in propositional logic that is not a clause by one or more equivalent statements that are clauses. For example, suppose we have a statement of the form $p \vee (q \wedge r)$. Because $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$, we can replace the single statement $p \vee (q \wedge r)$ by two statements $p \vee q$ and $p \vee r$, each of which is a clause. Using De Morgan's law, we can replace a statement of the form $\neg(p \vee q)$ by the two statements $\neg p$ and $\neg q$. We can also replace a conditional statement $p \longrightarrow q$ with the equivalent disjunction $\neg p \vee q$.

Example 1.6.10. Show that the premises $(p \wedge q) \vee r$ and $r \longrightarrow s$ imply the conclusion $p \vee s$.

Proof.

	Step	Reason
1.	$(p \wedge q) \vee r$	Premise
2.	$r \longrightarrow s$	Premise
3.	$(p \vee r) \wedge (q \vee r)$	1, Distributive law
4.	$\neg r \vee s$	2, Theorem 2
5.	$p \vee r$	3, Simplification
6.	$p \vee s$	5, 4, Resolution

□

1.6.4 Rules of Inference for Quantified Statements

We have discussed rules of inference for propositions. We will now describe some important rules of inference for statements involving quantifiers. These rules of inference are used extensively in mathematical arguments, often without being explicitly mentioned.

TABLE 2: Rules of Inference for Quantified Statements

Rule of Inference	**Name**
$\dfrac{(\forall x)P(x)}{\therefore P(c)}$	Universal instantiation
$\dfrac{P(c) \text{ for an arbitrary } c}{\therefore (\forall x)P(x)}$	Universal generalization
$\dfrac{(\exists x)P(x)}{\therefore P(c) \text{ for some element } c}$	Existential instantiation
$\dfrac{P(c) \text{ for some element } c}{\therefore (\exists x)P(x)}$	Existential generalization

Universal instantiation is the rule of inference used to conclude that $P(c)$ is true, where c is a particular member of the domain, given the premise $(\forall x)P(x)$. Universal

instantiation is used when we conclude from the statement "All women are wise" that "Aysha is wise," where Aysha is a member of the domain of all women.

Universal generalization is the rule of inference that states that $(\forall x)P(x)$ is true, given the premise that $P(c)$ is true for all elements c in the domain. Universal generalization is used when we show that $(\forall x)P(x)$ is true by taking an arbitrary element c from the domain and showing that $P(c)$ is true. The element c that we select must be an arbitrary, and not a specific, element of the domain. That is, when we assert from $(\forall x)P(x)$ the existence of an element c in the domain, we have no control over c and cannot make any other assumptions about c other than it comes from the domain. Universal generalization is used implicitly in many proofs in mathematics and is seldom mentioned explicitly. However, the error of adding unwarranted assumptions about the arbitrary element c when universal generalization is used is all too common in incorrect reasoning.

Existential instantiation is the rule that allows us to conclude that there is an element c in the domain for which $P(c)$ is true if we know that $(\exists x)P(x)$ is true. We cannot select an arbitrary value of c here, but rather it must be a c for which $P(c)$ is true. Usually we have no knowledge of what c is, only that it exists. Because it exists, we may give it a name (c) and continue our argument.

Existential generalization is the rule of inference that is used to conclude that $(\exists x)P(x)$ is true when a particular element c with $P(c)$ true is known. That is, if we know one element c in the domain for which $P(c)$ is true, then we know that $(\exists x)P(x)$ is true.

Exercises

Page 78: 1-12,19,20

Chapter 2

Basic Structures: Sets, Functions, Sequences,Sums, and Matrices

2.1 Sets

In this section, we study the fundamental discrete structure on which all other discrete structures are built, namely, the set. Sets are used to group objects together. Often, but not always, the objects in a set have similar properties. We now provide a defn of a set. This defn is an intuitive defn, which is not part of a formal theory of sets.

Definition 2.1.1. (Set)

(1) A set is an unordered collection of objects, called elements or members of the set. A set is said to contain its elements.

(2) If a is an element of a set A, then we write $a \in A$ (read: a is an element of A or a belongs to A).

(3) The note $a \notin A$ denotes that a is not an element of the set A.

Example 2.1.2.

(a) The set of all students in this class.

(b) The set of all moslems in the world.

(c) The set of all rational numbers less than 2.

(d) The set of all real solutions of the equation $x^2 + 1 = 0$.

(e) The set of all natural numbers between 5.5 and 8.2.

(f) The set of the numbers $1, 2, 3, 4$ and the letters a, b, c, d, e.

Example 2.1.3. Let $\mathbb{N}$ be the set of natural numbers. Then $1 \in \mathbb{N}$ but $-1 \notin \mathbb{N}$.

Important Sets

The following sets, each denoted using a boldface letter, play an important role in discrete mathematics:

- $\mathbb{N} = \{0, 1, 2, 3, \cdots\}$, the set of natural numbers.
- $\mathbb{Z} = \{\cdots, -2, -1, 0, 1, 2, \cdots\}$, the set of integers.
- $\mathbb{Z}^+ = \{1, 2, 3, \cdots\}$, the set of positive integers.
- $\mathbb{Q} = \{\frac{p}{q} \mid \ p \in \mathbb{Z}, q \in \mathbb{Z}, \text{ and } \ q \neq 0\}$, the set of rational numbers.
- $\mathbb{R}$, the set of real numbers.
- $\mathbb{R}^+$, the set of positive real numbers.
- $\mathbb{C}$, the set of complex numbers.

How to describe sets

Sets can be described by one of the following methods:

(1) By listing the elements between braces such as $\{a, b, c\}$ or $\{1, 2, \cdots\}$.

(2) By using the set builder note $\{x : p(x)\}$, where $p(x)$ is a propositional function describing the property that define the set.

Example 2.1.4.

(a) The set of all rational numbers less than 2 can be written as $\{x : \ x \in \mathbb{Q} \wedge x < 2\}$ or $\{\ x \in \mathbb{Q} : \ x < 2\}$.

(b) The set of all natural numbers between 2 and 3 can be written as $\{n : \ n \in \mathbb{N} \wedge \ 2 < n < 3\}$.

(c) The set of all real solutions of the equation $x^2 + 1 = 0$ can be written as $\{x : \ x \in \mathbb{R} \wedge \ x^2 + 1 = 0\}$.

Example 2.1.5. The set $\{\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}\}$ is a set containing four elements, each of which is a set. The four elements of this set are $\mathbb{N}$, the set of natural numbers; $\mathbb{Z}$, the set of integers; $\mathbb{Q}$, the set of rational numbers; and $\mathbb{R}$, the set of real numbers.

Remark 2.1.6. Note that the concept of a datatype, or type, in computer science is built upon the concept of a set. In particular, a datatype or type is the name of a set, together with a set of operations that can be performed on objects from that set. For example, boolean is the name of the set $\{0, 1\}$ together with operators on one or more elements of this set, such as AND, OR, and NOT.

Equality of Sets

Definition 2.1.7. (Equal sets)
Two sets A and B are said to be equal or identical if they contain the same elements. If the sets A and B are equal, then we write $A = B$.

Remark 2.1.8. $A = B$ means $(\forall x)[(x \in A) \leftrightarrow (x \in B)]$.

Example 2.1.9. Show that $A = \{a, b, c\}$ is equal to $B = \{b, c, a\}$.

Proof. A and B have the same elements. Thus $A = B$. □

Example 2.1.10. Let $A = \{1, 4, 1\}$ and $B = \{1, 4\}$. Show that $A = B$.

Proof. A and B have the same elements. Thus $A = B$. □

Remark 2.1.11. $a \neq \{a\}$.
a is an element but $\{a\}$ is a set containing one element.

The Empty Set

Definition 2.1.12. (Empty set)
The set that has no elements is called the empty set or null set, and it is denoted by $\emptyset$. The empty set can also be denoted by $\{\}$.

Remark 2.1.13. The statement $x \in \emptyset$ is false for every object x.

Example 2.1.14. $\{x : \ x \in \mathbb{R} \wedge \ x^2 + 1 = 0\} = \{x \in \mathbb{R} : \ x^2 + 1 = 0\} = \emptyset$.

Example 2.1.15. In each of the following, determine whether the statement is true or false.

(a) $x \in \{x\}$.

(b) $\{1\} = \{\{1\}\}$.

(c) $\{1, 2\} \in \{\{1, 2\}, 3\}$.

(d) $\phi = \{\phi\}$.

Proof.

(a) True.

(b) False.

(c) True.

(d) False.

□

2.1.1 Venn diagrams

To help in representing sets and sets operations graphically, we introduce Venn diagrams, named for the English mathematician and logician John Venn (1834-1923), which illustrate them. We shall represent the universal set U by a rectangle and we represent subsets of U by circles or other geometrical figures drawn inside the rectangle. Sometimes points are used to represent the particular elements of the set. Venn diagrams are often used to indicate the relationships between sets.

Although Venn diagrams seem much easier to use than proofs, a Venn diagram is no more than a visual aid, and is never a substitute for a real proof. Moreover, it is tricky to use Venn diagrams for more than three sets at a time, and this severely limits their use.

Example 2.1.16. Draw a Venn diagram for the set $A = \{1, 2, 3, 4\}$.

Proof. □

2.1.2 Subsets

It is common to encounter situations where the elements of one set are also the elements of a second set. We now introduce some terminology and note to express such relationships between sets.

Definition 2.1.17. (Subset)
Let A and B be two sets.

(1) If every element of A is an element of B, then we say that A is a subset of B and we write $A \subseteq B$ or $B \supseteq A$.

(2) If A is a subset of B, then we say that B is a superset of A.

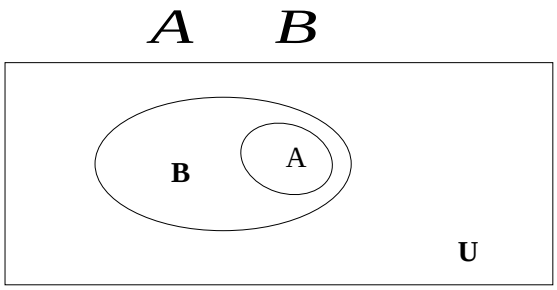
Remarks 2.1.18.

(1) $A \subseteq B$ means $(\forall x)[(x \in A) \to (x \in B)]$.

(2) $(A = B) \Leftrightarrow [(A \subseteq B) \wedge (B \subseteq A)]$.

Example 2.1.19. Let $A = \{1, 2, a\}$ and let $B = \{a, b, 1, 2, 3\}$. Then $A \subseteq B$.

Example 2.1.20. Let $A = \{1, 2, 3\}$ and let $B = \{3, 1, 2\}$. Then $A \subseteq B$.



Example 2.1.21. In each of the following, determine whether the statement is true or false. If it is true, prove it. If it is false, disprove it by an example (counterexample).

(a) If $x \in A$ and $A \in B$, then $x \in B$.

(b) If $x \in B$ and $A \subseteq B$, then $x \notin A$.

Proof.

(a) False.

Let $A = \{x\}$ and let $B = \{\{x\}\}$. Then $x \in A,\ A \in B$, but $x \notin B$.

(b) False.

Let $A = \{1, 2\}$ and $B = \{1, 2, 3\}$. Then $A \subseteq B$ and $1 \in B$ and $1 \in A$.

□

Theorem 1 shows that every nonempty set S is guaranteed to have at least two subsets, the empty set and the set S itself, that is, $\emptyset \subseteq S$ and $S \subseteq S$.

Theorem 2.1.22. *Let S be any set. Then*

(1) $\emptyset \subseteq S$.

(2) $S \subseteq S$.

Proof. Let S be any set.

(1) We want to show that $\emptyset \subseteq S$; that is

$$(\forall x)[(x \in \emptyset) \to (x \in S)].$$

In other words, we want to show that the open sentence $(x \in \phi) \to (x \in S)$ is true for every x.

We know that $x \in \emptyset$ is false for every x. Therefore $(x \in \emptyset) \to (x \in S)$ is true for every x.

(2) We want to show that $S \subseteq S$; that is

$$(\forall x)[(x \in S) \to (x \in S)].$$

□

Example 2.1.23. Which of the following statements is **false** and which is **true**:
(a) $\emptyset \subset \{\emptyset\}$ (b) $\emptyset \in \{\emptyset\}$
(c) $\{\emptyset\} \subset \emptyset$ (d) $\emptyset = \{\emptyset\}$

Proof. □

When we wish to emphasize that a set A is a subset of a set B but that $A \neq B$, we use the concept of prober subset.

Definition 2.1.24. (Proper subset)

(1) We say that a set A is a proper subset of a set B, and write $A \subset B$, if $A \subseteq B$ and $A \neq B$.

(2) If $A \subset B$, then B is called a proper superset of A.

Example 2.1.25. $\mathbb{Z} \subset \mathbb{Q}$.

Example 2.1.26. If $A = \{-1, 0, 1\}$ and $B = \{1, -1, 0\}$, then $A \subseteq B$ but A is not a proper subset of B.

2.1.3 The Size of a Set

Sets are used extensively in counting problems, and for such applications we need to discuss the sizes of sets.

Definition 2.1.27. Let S be a set.

(a) If there are exactly n distinct elements in S where n is a nonnegative integer, we say that S is a finite set and that n is the cardinality of S.

(b) The cardinality of S is denoted by $|S|$.

(c) The cardinality of the empty set is $|\emptyset| = 0$.

Example 2.1.28.

(a) The set of all moslems in the world is a finite set.

(b) Let A be the set of positive integers less than 5 Then A is a finite set and $|A| = 4$.

(c) Let S be the set of letters in the English alphabet. Then $|S| = 26$.

Definition 2.1.29. (Infinite set)
A set is said to be infinite if it is not finite.

Example 2.1.30. The set of all rational numbers $\mathbb{Q}$ is an infinite set.

Power Sets

Many problems involve testing all combinations of elements of a set to see if they satisfy some property. To consider all such combinations of elements of a set S, we build a new set that has as its members all the subsets of S.

Definition 2.1.31. (Power set)
Let S be a set. The power set of S is the set of subsets of S and it is denoted by $\wp(S)$.

Remarks 2.1.32.

(1) $\wp(S) = \{A : \ A \subseteq S\}$.

(2) For any set S, $S \in \wp(S)$ and $\emptyset \in \wp(S)$.

Example 2.1.33. Let $S = \{1\}$. Find the power set of S.

Proof. $\wp(S) = \{\emptyset, S\}$ □

Example 2.1.34. Find the power set of $S = \{a, b, c\}$.

Proof. $\wp(S) = \{\emptyset, S, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}\}$ □

Example 2.1.35. Find the power set of $\emptyset$.

Proof. $\wp(\emptyset) = \{\emptyset\}$ □

2.1.4 Cartesian Products

The order of elements in a collection is often important. Because sets are unordered, a different structure is needed to represent ordered collections. This is provided by ordered $n-$tuples.

Definition 2.1.36. (Ordered $n-$tuples)

(1) Given any n objects $a_1, \ a_2, \cdots, a_n$, the object $(a_1, \ a_2, \cdots, a_n)$ is called the ordered $n-$tuples of $a_1, \ a_2, \cdots, a_n$.

(2) If $(a_1, \ a_2, \cdots, a_n)$ is an ordered $n-$tuples, then a_1 is called the first coordinate, a_2 is called the second coordinate,$\cdots$, and a_n is called $n-$th coordinate.

(3) We say that two ordered $n-$tuples $(a_1, \ a_2, \cdots, a_n)$ and $(b_1, \ b_2, \cdots, b_n)$ are equal, and write $(a_1, \ a_2, \cdots, a_n) = (b_1, \ b_2, \cdots, b_n)$, if and only if $a_1 = b_1$, $a_2 = b_2$, $\cdots$, $a_n = b_n$.

Remark 2.1.37.

(i) The adjective "ordered" here emphasizes that the order in which the objects appear in an $n-$tuples $(a_1, \ a_2, \cdots, a_n)$ is essential.

(ii) The ordered $2-$tuples are called ordered pairs. The two ordered pairs (a, b) and (c, d) are equal if and only if $a = c$ and $b = d$.

Example 2.1.38. If $(1,2)$ and $(2,1)$ are two ordered pairs, then $(1,2) \neq (2,1)$.

Example 2.1.39. $(x,y) = (2,3)$ if and only if $x = 2$ and $y = 3$.

Definition 2.1.40. (Cartesian products)
Let A and B be sets. The cartesian product of A and B, denoted by $A \times B$, is the set of all ordered pairs (x,y) with $x \in A$ and $y \in B$. In symbols

$$A \times B = \{(x,y) : \ x \in A \wedge y \in B\}.$$

Example 2.1.41. Let $A = \{1,2,3\}$ and let $B = \{a,b\}$. Find $A \times B$ and $B \times A$.

Proof. $A \times B = \{(x,y) : \ x \in A \wedge y \in B\} = \{(1,a),(1,b),(2,a),(2,b),(3,a),(3,b)\}$. □

Example 2.1.42. Let A be any set. Find $A \times \emptyset$ and $\emptyset \times A$.

Proof. $A \times \emptyset = \{(x,y) : \ x \in A \wedge y \in \emptyset\} = \emptyset$ (since $y \in \emptyset$ is false). □

Definition 2.1.43. The Cartesian product of the sets $A_1, \ A_2, \cdots, \ A_n$, denoted by $A_1 \times A_2 \times \cdots \times A_n$, is the set of ordered $n-$tuples $(a_1, \ a_2, \cdots, a_n)$, where a_i belongs to A_i for $i = 1, 2, \cdots, n$. In other words,

$$A_1 \times A_2 \times \cdots \times A_n = \{(a_1, \ a_2, \cdots, a_n) | \quad a_i \in A_i \ \text{ for } \ i = 1, 2, \cdots, n\}.$$

Example 2.1.44. Find $A \times B \times C$ if $A = \{1,2\}$, $B = \{0,2,4\}$, and $C = \{1,3\}$.

Proof. □

Remark 2.1.45. We use the note A^2 to denote $A \times A$, the Cartesian product of the set A with itself. Similarly, $A^3 = A \times A \times A$, $A^4 = A \times A \times A \times A$, and so on.

2.1.5 Relations

Given sets A and B, not necessarily distinct, when we say that an element a of A is related to another element b of B by a relation R we are making a statement about the ordered pair (a,b) in the Cartesian product $A \times B$. Therefore, a mathematical defn of a relation can be precisely given in term of ordered pairs in Cartesian product of sets.

Definition 2.1.46. (Relations)
A relation R from a set A to a set B is a subset of the Cartesian product $A \times B$.

Notation 2.1.47. If R is a relation and $(a,b) \in R$, then we write $a \ R \ b$ (read: a is $R-$related to b).

Example 2.1.48. Let $A = \{a,b\}$ and let $B = \{1,-1\}$. Then $R = \{(a,1),(b,1)\}$ is a relation from A to B.

Example 2.1.49. Let $A = \{1,2,3\}$ and let $R = \{(1,1),(2,2),(1,3)\}$. Then R is a relation from A to A.

Remark 2.1.50. If R is a relation from A to A, then R is called a relation on A.

Exercise

Page 125: 1-21, 27, 32-38,

2.2 Set Operations

Two, or more, sets can be combined in many different ways. For instance, starting with the set of mathematics majors at your school and the set of computer science majors at your school, we can form the set of students who are mathematics majors or computer science majors, the set of students who are joint majors in mathematics and computer science, the set of all students not majoring in mathematics, and so on.

In set theory, there are three operations: union, intersection, and difference.

Definition 2.2.1. (Union)
The union of any two sets A and B, denoted by $A \cup B$, is the set of all elements x such that x belongs to at least one of the two sets A and B. In symbols,

$$A \cup B = \{x : \ (x \in A) \vee (x \in B)\}.$$

Example 2.2.2. Let $A = \{1, 2, 3, 4, 5\}$ and let $B = \{1, 3, 5, 7\}$. Find $A \cup B$.

Proof. □

Example 2.2.3. Find $\mathbb{R} \cup \mathbb{N}$.

Proof. □

Definition 2.2.4. (Intersection)
The intersection of any two sets A and B, denoted by $A \cap B$, is the set of all elements x such that x belongs to both A and B. In symbols,

$$A \cap B = \{x : \ (x \in A) \wedge (x \in B)\} = \{x \in A : \ x \in B\}.$$

Example 2.2.5. Let $A = \{1, 2, 3, 4, 5\}$ and let $B = \{1, 3, 5, 7\}$. Find $A \cap B$.

Proof. □

Example 2.2.6. Let $I = [-1, 0] = \{x \in \mathbb{R} : \quad -1 \leq x \leq 10\}$. Find $I \cap \mathbb{Z}$ and $I \cap \mathbb{N}$.

Proof. $I \cap \mathbb{Z} = \{0, -1\}$ and $I \cap \mathbb{N} = \{0\}$. □

Definition 2.2.7. (Disjoint sets)
Let A and B be two sets. If $A \cap B = \emptyset$, then A and B are said to be disjoint.

Cardinality of $A \cup B$

We are often interested in finding the cardinality of a union of two finite sets A and B. Note that $|A| + |B|$ counts each element that is in A but not in B or in B but not in A exactly once, and each element that is in both A and B exactly twice. Thus, if the number of elements that are in both A and B is subtracted from $|A| + |B|$, elements in $A \cap B$ will be counted only once. Hence,

$$|A \cup B| = |A| + |B| - |A \cap B|.$$

The generalization of this result to unions of an arbitrary number of sets is called the principle of inclusion-exclusion. The principle of inclusion-exclusion is an important technique used in enumeration. We will discuss this principle and other counting techniques in detail in Chapters 6 and 8.

2.2.1 The difference of two sets

Definition 2.2.8. Let A and B be sets. The difference of A and B is the set $A - B$ defined by

$$A - B = \{x \in A : \ x \notin B\} = \{x : \ x \in A \wedge x \notin B\}.$$

Remarks 2.2.9.

(1) In the above definition it is NOT assumed that $B \subseteq A$.

(2) The difference of of A and B is also called the complement of B with respect to A.

Example 2.2.10. Let $A = \{a, 1, \mathbb{R}\}$ and $B = \{1, 2, 3, 4\}$. Find $A - B$ and $B - A$.

Proof. □

Example 2.2.11. Let $A = \{x, 1, y, -1\}$ and $B = \{0, 1, 2, x, y\}$. Find $B - (A \cup B)$.

Proof. $\emptyset$ □

Definition 2.2.12. (The complement)
If U is the universal set, then the complement of A is $\bar{A} = U - A$.

Remark 2.2.13. Since $\bar{A} = U - A$, $x \in \bar{A} \equiv x \notin A$.

Example 2.2.14. Let $U = \mathbb{R}$ and $A = (-\infty, 0]$. Find $\overline{A}$.

Proof. □

2.2.2 Set Identities

The following theorems lists the most important set identities. We will prove several of these identities here, using three different methods. These methods are presented to illustrate that there are often many different approaches to the solution of a problem. The proofs of the remaining identities will be left as exercises. The reader should note the similarity between these set identities and the logical equivalences discussed in Section 1.3. In fact, the set identities given can be proved directly from the corresponding logical equivalences. Furthermore, both are special cases of identities that hold for Boolean algebra (discussed in Chapter 12). One way to show that two sets are equal is to show that each is a subset of the other. Recall that to show that one set is a subset of a second set, we can show that if an element belongs to the first set, then it must also belong to the second set. We generally use a direct proof to do this. We illustrate this type of proof by establishing the first of De Morgan's laws.

Theorem 2.2.15. *(Set Identities)*
Let U be the universal set and let A, B, and C be subsets of U. Then we have:

(1) Identity laws: $A \cup \emptyset = A$ *and* $A \cap U = A$.

(2) Domination laws: $A \cap \emptyset = \emptyset$ *and* $A \cup U = U$.

(3) Idempotent laws: $A \cup A = A$ *and* $A \cap A = A$.

(4) Complementation law: $\overline{\overline{A}} = A$.

(5) The commutative laws : $A \cup B = B \cup A$ *and* $A \cap B = B \cap A$.

(6) The associative laws: $A \cup (B \cup C) = (A \cup B) \cup C$ *and* $A \cap (B \cap C) = (A \cap B) \cap C$.

(7) The distributive laws: $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ *and* $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

(8) De Morgan's laws: $\overline{(A \cup B)} = \overline{A} \cap \overline{B}$ *and* $\overline{(A \cap B)} = \overline{A} \cup \overline{B}$.

(9) Absorption laws: $A \cup (A \cap B) = A$ *and* $A \cap (A \cup B) = A$.

(10) Complement laws: $A \cap \overline{A} = \emptyset$ *and* $A \cup \overline{A} = U$.

2.2.3 Mathematical Proofs

Proofs are essential in mathematical reasoning because they demonstrate that the conclusions are true. Generally speaking, a mathematical explanation for a conclusion has no value if the explanation cannot be backed up by an acceptable proof.

A proof is a complete justification of the truth of a statement called a theorem. It generally begins with some hypotheses stated in the theorem and proceeds by correct reasoning to the claimed statement. It is nothing more than an argument that presents a line of reasoning explaining why the statement follows from known facts.

Basic Proof Rules

The following four rules provide guidance about what statements are allowed in a proof, and when.

(1) At any time state an assumption, an axiom, or a previously proved result.

(2) The replacement rule: At any time state a statement equivalent to any statement earlier in the proof.

(3) The tautology rule: At any time state a statement whose symbolic translation is a tautology.

(4) The modus ponens rule: At any time after p and $p \to q$ appear in a proof, state that q is true.

Remark 2.2.16. The modus ponens rule is the most fundamental rule of reasoning. It is based on the tautology

$$[(p \to q) \wedge p] \to q.$$

Methods of Mathematical Proof

There are several methods of proof. Here we introduce three basic methods of proof.

- **(1) DIRECT PROOF:** A direct proof is a logical step-by-step argument from the given conditions to the conclusion.
- **(2) PROOF BY CONTRADICTION:** A proof by contradiction is an indirect proof method. It is based on the tautology $p \longleftrightarrow [\neg p \longrightarrow (\neg q \wedge q)]$. That is, to prove a statement p is true, we prove that the statement $\neg p \longrightarrow (\neg q \wedge q)$ is true for some statement q.

  The logic behind such a proof is that if a statement cannot be false, then it must be true. Thus, to prove by contradiction that a statement p is true, we temporarily assume that p is false and then see what would happen. If what happens is an impossibility-that is, a contradiction-then we know that p must be true.
- **(3) PROOF BY MATHEMATICAL INDUCTION:** Proof by mathematical induction is a very useful method in proving the validity of a mathematical statement $(\forall n)P(n)$ involving integers n greater than or equal to some initial integer n_0.

Advances for writing proofs

There is no unified method to write a mathematical proof. However there are certain techniques that are often useful when writing proofs. Here are some advances that may help in writing correct proofs.

How to start

Begin a proof by rewriting what you are given and what you are asked to prove in a more convenient form. Often this involves converting word to symbols and utilizing the definitions of the terms used in the statements.

Justify each step

As a general rule, when you write a step in a proof, ask yourself if deducing that step is valid in the sense that it uses one of the four basic proof rules above. If the step follows as a result of the use of a tautology, it is not necessary to cite the tautology in your proof. In fact, with practice you should eventually come to write proofs without purposefully thinking about tautologies. What is necessary is that every step be justifiable.

Example 2.2.17. Prove that $A \cup \emptyset = A$.

Proof.

$$\begin{array}{lll} (A \cup \emptyset) & = \{x : \ (x \in A) \vee (x \in \emptyset)\} & \text{(definition of } \cup) \\ & = \{x : \ x \in A \vee F\} & (x \in \emptyset \equiv F) \\ & = \{x : \ x \in A\} & (p \vee F \equiv p) \\ & = A & \end{array}$$

□

Example 2.2.18. Prove that $A \cap A = A$.

Proof.

$$\begin{array}{lll} (A \cap A) & = \{x : \ (x \in A) \wedge (x \in A)\} & \text{(definition of } \cap) \\ & = \{x : \ x \in A\} & (p \wedge p \equiv p) \\ & = A & \end{array}$$

□

Example 2.2.19. Prove that $A \cup (B \cup C) = (A \cup B) \cup C$.

Proof.

$$\begin{array}{lll} A \cup (B \cup C)] & = \{x : \ (x \in A) \vee (x \in B \vee x \in C)\} & \text{(definition of } \cup) \\ & = \{x : \ (x \in A \vee x \in B) \vee x \in C)\} & \text{(associative law for } \vee) \\ & = (A \cup B) \cup C & \text{(definition of } \cup) \end{array}$$

□

Example 2.2.20. Prove that $\overline{\overline{A}} = A$.

Proof.

$$\begin{array}{lll} \overline{\overline{A}} & = \{x : \neg(x \in \overline{A})\} & \text{(definition of complement)} \\ & = \{x : \neg(\neg x \in A)\} & \text{(definition of complement)} \\ & = \{x : x \in A\} & (\neg\neg p \equiv p) \\ & = A & \end{array}$$

□

2.2.4 Proving conditional statements $p \longrightarrow q$ (conditional proof)

The most famous example is the direct proof of statements of the form $p \longrightarrow q$ which proceeds in a step-by-step fashion from the condition p to the conclusion q. Since $p \longrightarrow q$ is

false only when p is true and q is false, it suffices to show that this situation cannot happen. The direct way to proceed is to assume that p is true and show (deduce) that q is also true.

A direct proof of $p \longrightarrow q$ will have the following form:

Direct proof of $p \longrightarrow q$:
Assume p.
$\vdots$
Therefore, q.

2.2.5 Proofs Involving Quantifiers

Now we discuss specifically the proof methods for statements with quantifiers.

2.2.6 (1) Proving $(\forall x)P(x)$

To prove a statement of the form $(\forall x)P(x)$, we must show that $P(x)$ is true for every object x in the universe. A direct proof is begun by letting x represent an arbitrary object in the universe, and then showing that $P(x)$ is true for that object. In the proof we may use only properties of x that are shared by every element of the universe. Then, since x is arbitrary, we can conclude that is $(\forall x)P(x)$ true.

Thus a direct proof of $(\forall x)P(x)$ has the following form:

Direct proof of $(\forall x)P(x)$
Let x be an arbitrary object in the universe.
(The universe should be named or its objects should be described.)
$\vdots$
Hence $P(x)$ is true.
Since x is arbitrary, $(\forall x)P(x)$ is true.

Example 2.2.21. Prove that $A \subseteq C$ and $B \subseteq C$ implies $(A \cup B) \subseteq C$.

Proof. Assume $A \subseteq C$ and $B \subseteq C$. We want to prove that

$$(A \cup B) \subseteq C \equiv (\forall x)[x \in (A \cup B) \rightarrow x \in C].$$

Let x be any element.
Assume that $x \in (A \cup B)$.
Then $x \in A \vee x \in B$.
This implies $x \in C \vee x \in C$.
Therefore, $x \in C$. □

2.2.7 Proving biconditional statements $p \longleftrightarrow q$

Proofs of biconditional statements $p \longleftrightarrow q$ often make use of the tautology

$$(p \longleftrightarrow q) \equiv (p \longrightarrow q) \wedge (q \longrightarrow p).$$

Proofs of $p \longleftrightarrow q$ generally have the following two-part form:

Two-Part Proof Of $p \longleftrightarrow q$
(i) Show $p \longrightarrow q$.
(ii) Show $q \longrightarrow p$.
Therefore, $p \longleftrightarrow q$.

In some cases it is possible to prove a biconditional statement $p \Longleftrightarrow q$ that uses the connective throughout. This amounts to starting with p and then replacing it with a sequence of equivalent statements, the last one being q. With n intermediate statements $R_1, R_2, \ldots, R_n$, a biconditional proof of $p \Longleftrightarrow q$ has the form:

Biconditional Proof Of $p \longleftrightarrow q$
$p \longleftrightarrow R_1$
$\longleftrightarrow R_2$
$\vdots$
$\longleftrightarrow R_n$
$\longleftrightarrow q.$

Example 2.2.22. Let A and B be two sets. Prove that $A \cap B = A$ if and only if $A \cup B = B$.

Proof. We will use two parts proof.

(1) $(A \cap B = A) \Rightarrow (A \cup B = B)$?
Assume $A \cap B = A$. Then

$A \cup B$	$= (A \cap B) \cup B$	$(A \cap B = A)$
	$= (A \cup B) \cap (B \cup B)$	(distributive law)
	$= (A \cup B) \cap B$	$(B \cup B = B)$
	$= B$	$(B \subseteq A \cup B)$

(2) $(A \cup B = B) \Rightarrow (A \cap B = A)$?
Assume $A \cup B = B$. Then

$A \cap B$	$= A \cap (A \cup B)$	
	$= (A \cap A) \cup (A \cap B)$	(distributive law)
	$= A \cup (A \cap B)$	$(A \cap A = A)$
	$= A$	$(A \cap B \subseteq A)$

□

2.2.8 When to use proof by contradiction

Proof by contradiction is a natural way to proceed when negating the conclusion gives you something concrete to manipulate.

A proof by contradiction has the following form:

Proof of p by contradiction:
Assume $\sim p$.
$\vdots$
Therefore, q.
$\vdots$
Therefore, $\sim q$.
Hence, $q \wedge \sim q$ a contradiction.

Remark 2.2.23. Two aspects about proofs by contradiction are especially noteworthy. First, this method of proof can be applied to any proposition p, whereas direct proofs and proofs by contraposition can be used only for conditional statements. Second, the proposition q does not appear on the left side of the tautology $\sim p \longrightarrow (q \wedge \sim q)$. The strategy of proving p by proving $\sim p \longrightarrow (q \wedge \sim q)$ then, has an advantage and a disadvantage. We do not know what proposition to use for q, but any proposition that will do the job is a good one. This means a proof by contradiction may require a spark of insight to determine a useful q.

Example 2.2.24. Prove that $A \cap \overline{A} = \emptyset$.

Proof. Assume $A \cap \overline{A} \neq \emptyset$ (indirect proof)
Then there exists $x \in A \cap \overline{A}$ (definition of $\emptyset$).
$x \in A \cap \overline{A} \equiv (x \in A) \wedge (x \in \overline{A})$ (definition of $\cap$).
$\equiv (x \in A) \wedge [x \in U \wedge x \notin A]$ (definition of complement).
$\equiv F \wedge x \in U$ ($p \wedge \neg p \equiv F$).
$\equiv F$ ($F \wedge p \equiv F$). □

2.2.9 Membership Tables

Set identities can also be proved using membership tables. We consider each combination of sets that an element can belong to and verify that elements in the same combinations of sets belong to both the sets in the identity. To indicate that an element is in a set, a 1 is used; to indicate that an element is not in a set, a 0 is used. (The reader should note the similarity between membership tables and truth tables.)

Example 2.2.25. Use a membership table to show that $A \cap \overline{A} = \emptyset$.

Proof.

A	$\overline{A}$	$A \cap \overline{A}$
1	0	0
0	1	0

□

Example 2.2.26. Use a membership table to show that $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$.

Proof.

A	B	C	$B \cap C$	$A \cup (B \cap C)$	$A \cup B$	$A \cup C$	$(A \cup B) \cap (A \cup C)$
1	1	1	1	1	1	1	1
1	1	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	0	0	0	1	1	1	1
0	1	1	1	1	1	1	1
0	1	0	0	0	1	0	0
0	0	1	0	0	0	1	0
0	0	0	0	0	0	0	0

□

Generalized Unions and Intersections

Because unions and intersections of sets satisfy associative laws, the sets $A \cup B \cup C$ and $A \cap B \cap C$ are well defined. That is, the meaning of this notation is unambiguous when A, B, and C are sets. In other words, we do not have to use parentheses to indicate which operation comes first because $(A \cup B) \cup C = A \cup (B \cup C)$ and $(A \cap B) \cap C = (A \cap B) \cap C$. Note that $A \cap B \cap C$ contains those elements that are in all of A, B, and C, and $A \cup B \cup C$ contains those elements that are in at least one of the sets A, B, and C.

Example 2.2.27. Let $A = \{1, 2\}$, $B = \{1, 3\}$, $C = \{1, 3, 4, 5\}$. Find $A \cup B \cup C$ and $A \cap B \cap C$.

Proof. $A \cap B \cap C = \{1\}$ and $A \cup B \cup C = \{1, 2, 3, 4, 5\}$. □

Exercises

Page 136: 1-34

2.3 Functions

The concept of function is one of the most basic ideas in every branch of mathematics. For example, in discrete mathematics functions are used in the definition of such discrete structures as sequences and strings. Functions are also used to represent how long it takes a computer to solve problems of a given size. Many computer programs and subroutines are designed to calculate values of functions. Recursive functions, which are functions defined in terms of themselves, are used throughout computer science; they will be studied in Chapter 5. This section reviews the basic concepts involving functions needed in discrete mathematics.

Definition 2.3.1. (Function)
Let A and B be nonempty sets. A function f from A to B is an assignment of exactly one element of B to each element of A.

Notation 2.3.2.

(1) If f is a function from A to B, then we write $f : A \longrightarrow B$.

(2) We write $f(a) = b$ if b is the unique element of B assigned by the function f to the element a of A.

Remark 2.3.3. A function $f : A \longrightarrow B$ can also be defined in terms of a relation from A to B. Recall from Section 2.1 that a relation from A to B is just a subset of $A \times B$. A relation from A to B that contains one, and only one, ordered pair (a, b) for every element $a \in A$, defines a function f from A to B. This function is defined by the assignment $f(a) = b$, where (a, b) is the unique ordered pair in the relation that has a as its first element.

Example 2.3.4. Let $A = \{1, -1\}, B = \{0\}$, and $f = \{(1, 0), (-1, 0)\}$. Is f a function?

Proof. $Dom(f) = A$ and only $(1, 0) \in f$ and only $(-1, 0) \in f$. Thus f is a function. □

Example 2.3.5. Let $A = \{1, 2, 3\}, B = \{5, 10, 15, 20\}$, and $f = \{(1, 5), (2, 20)\}$. Is f a function?

Proof. Since $Dom(f) = \{1, 2\} \neq A$, f is NOT a function. □

Example 2.3.6. Let $A = \{1, 2, 3\}, B = \{5, 10, 15, 20\}$, and $g = \{(1, 5), (2, 20), (3, 15), (3, 10)\}$. Is g a function?

Proof. g is NOT a function since $(3, 15) \in g$ and $(3, 10) \in g$, but $10 \neq 15$. □

Remark 2.3.7. Functions are sometimes also called mappings or transformations.

Definition 2.3.8. Let $f : A \longrightarrow B$ be a function.

(1) We say that A is the domain of f and B is the codomain of f.

(2) If $b = f(a)$, then we say that b is the image of a under f and a is a pre-image of b under f.

(3) The range, or image, of f is the set of all images of elements of A.

Example 2.3.9. Let R be the relation with ordered pairs $(Ahmed, 22)$, $(Ali, 24)$, $(Mohammed, 21)$, $(Aysha, 22)$, $(Fatma, 24)$, and $(Omer, 22)$. Here each pair consists of a graduate student and this student's age. Specify a function f determined by this relation and find the domain, codomain and range of f.

Proof. If f is a function specified by R, then $f(Ahmed) = 22$, $f(Ali) = 24$, $f(Mohammed) = 21$, $f(Aysha) = 22$, $f(Fatma) = 24$, and $f(Omer) = 22$. (Here, $f(x)$ is the age of x , where x is a student.) The domain of f is the set $\{Ahmed, Ali, Mohammed, Aysha, Fatma, Omer\}$. The codomain of f is the set $\{21, 22, 24\}$. □

Example 2.3.10. Let f be the function that assigns the last two bits of a bit string of length 2 or greater to that string. For example, $f(11010) = 10$. Then, the domain of f is the set of all bit strings of length 2 or greater, and both the codomain and range are the set $\{00, 01, 10, 11\}$.

Example 2.3.11. Let $f : \mathbb{R} \longrightarrow \mathbb{R}$ be defined by $f(x) = x^2 + 1$. Find the domain, codomain and range of f.

Proof. The domain of $f = \mathbb{R}$, the codomain of $f = \mathbb{R}$ and the range of $f = \{x \in \mathbb{R} : x \geq 1\} = [1, \infty)$. □

Remarks 2.3.12.

(1) A function is called real-valued if its codomain is the set of real numbers.

(2) A function is called integer-valued if its codomain is the set of integers.

Definition 2.3.13. (Addition and multiplication of functions)
Let f_1 and f_2 be functions from A to $\mathbb{R}$. Then $f_1 + f_2$ and $f_1 f_2$ are also functions from A to $\mathbb{R}$ defined for all $x \in A$ by

$$(f_1 + f_2)(x) = f_1(x) + f_2(x),$$
$$(f_1 f_2)(x) = f_1(x) f_2(x).$$

2.3.1 One-to-One and Onto Functions

Definition 2.3.14. (One-to-One Function)

(1) A function $f : A \to B$ is said to be injective or one-to-one provided that if $x_1, x_2 \in A$ with $f(x_1) = f(x_2)$, then $x_1 = x_2$.

(2) An injective function is called an injunction.

Remark 2.3.15. $f : A \to B$ is injective $\equiv (\forall x_1)(\forall x_2)[f(x_1) = f(x_2) \to x_1 = x_2]$
$\equiv (\forall x_1)(\forall x_2)[x_1 \neq x_2 \to f(x_1) \neq f(x_2)]$

Example 2.3.16. Consider the function f from $A = \{a, b, c, d\}$ to $B = \{1, 2, 3, 4, 5\}$ be given by $f(a) = 4$, $f(b) = 5$, $f(c) = 1$, and $f(d) = 3$. Determine whether f is one-to-one.

Proof. The function f is one-to-one because f takes on different values at the four elements of its domain. □

Example 2.3.17. Let $f : \mathbb{Z} \to \mathbb{N}$ be a function given by $f(n) = n^2$. Show that f is not one-to-one.

Proof. $f(1) = f(-1)$ but $1 \neq -1$. □

Example 2.3.18. Suppose that each student in a class is assigned one problem from a set of exercises, each to be done by a single student. In this situation, the function f that assigns a problem to each student is one-to-one. To see this, note that if x and y are two different students, then $f(x) \neq f(y)$ because the two students x and y must be assigned different problems.

2.3.2 Increasing and decreasing functions

We now give some conditions that guarantee that a function is one-to-one.

Definition 2.3.19. Let f be a function whose domain A and codomain B are subsets of the set of real numbers. Let $x, y \in A$ with $x < y$.

(1) f is called increasing if $f(x) \leq f(y)$.

(2) f is called strictly increasing if $f(x) < f(y)$.

(3) f is called decreasing if $f(x) \geq f(y)$.

(4) f is called strictly decreasing if $f(x) > f(y)$.

Example 2.3.20. Show that a function that is either strictly increasing or strictly decreasing must be one-to-one.

Proof. Let $f : A \to B$ be a strictly increasing function and let x_1 and x_2 be two elements in A such that $x_1 \neq x_2$. If $x_1 < x_2$, then $f(x_1) < f(x_2)$. If $x_1 > x_2$, then $f(x_1) > f(x_2)$. Thus $f(x_1) \neq f(x_2)$ and hence f is one-to-one. □

Definition 2.3.21. (Onto)

(1) A function $f : A \to B$ is said to be onto or surjective provided that $\forall y \in B$ there exists at least one $x \in A$ such that $f(x) = y$. In other words, $f : A \to B$ is onto if and only if every member of the codomain is the image of some element of the domain.

(2) A surjective function is called a surjection.

Remark 2.3.22. $f : A \to B$ is onto $\equiv (\forall y \in B)(\exists x \in A)(f(x) = y)$.

Example 2.3.23. Let $f : \mathbb{R} \to [0, \infty)$ be a function given by $f(x) = x^2$. Show that f is an onto function.

Proof. Let $y \in [0, 1)$ and let $x = \sqrt{y}$. Then $x \in \mathbb{R}$ and $f(x) = y$. Thus f is surjective. □

Example 2.3.24. Let $f : \mathbb{R} \to \mathbb{R}$ be a function given by $f(x) = x^2$. Then f is not an onto function since $f(x) \neq -1$ for all $x \in \mathbb{R}$.

Definition 2.3.25. (Bijection)

(1) A function $f : A \to B$ is said to be bijective if it is both one-to-one and onto.

(2) A bijection is also called a one-to-one correspondence.

Example 2.3.26. Let f be the function from $A = \{a, b, c, \}$ to $B = \{1, 2, 3\}$ with $f(a) = 3$, $f(b) = 1$, and $f(c) = 2$. Is f a bijection?

Proof. The function f is one-to-one and onto. It is one-to-one because no two values in the domain are assigned the same function value. It is onto because all four elements of the codomain are images of elements in the domain. Hence, f is a bijection. □

Example 2.3.27. Let $A = \{0, 1, 2, 3\}$ and let $B = \{4, 5, 6, 7\}$.

(a) Give a function f from A to B such that f is not a one-to-one correspondence.

(b) Give a function g from A to B such that g is a one-to-one correspondence.

Proof. (a) Let $f : A \to B$ be given by $f(0) = f(1) = f(2) = f(3) = 4$. Then f is a function from A to B and it is not a one-to-one correspondence.

(b) Let $g : A \to B$ be given by $g(0) = 4,\ g(1) = 5,\ g(2) = 6,\ g(3) = 7$. Then g is a one-to-one correspondence.

□

2.3.3 Inverse Functions

Now consider a one-to-one correspondence f from the set A to the set B. Because f is an onto function, every element of B is the image of some element in A. Furthermore, because f is also a one-to-one function, every element of B is the image of a unique element of A. Consequently, we can define a new function from B to A that reverses the correspondence given by f.

Definition 2.3.28. (Inverse function)
Let f be a one-to-one correspondence from the set A to the set B. The inverse function of f is the function that assigns to an element b belonging to B the unique element a in A such that $f(a) = b$. The inverse function of f is denoted by f^{-1}. Hence, $f^{-1}(b) = a$ when $f(a) = b$.

Remark 2.3.29. Be sure not to confuse the function f^{-1} with the function $1/f$, which is the function that assigns to each x in the domain the value $1/f(x)$.

Example 2.3.30. Let $A = \{1, 2\}$, $B = \{\pi, e\}$, and let $f : A \to B$ be given by $f(1) = \pi,\ f(2) = e$. Is f invertible, and if it is, what is its inverse?

Proof. Since f is a one-to-one correspondence, it is invertible. The inverse of f is $f^{-1} : B \to A$ given by $f^{-1}(\pi) = 1$ and $f^{-1}(e) = 2$. □

Example 2.3.31. Let $f : [0, 1) \to [0, 1)$ be given by $f(x) = x^2$. Show that f is invertible and find its inverse.

Proof.

(1) f is one-to-one?

Let $x_1, x_2 \in [0, 1)$ such that $f(x_1) = f(x_2)$. Then we have $x_1^2 = x_2^2$. It follows that $x_1 = x_2$ or $x_1 = -x_2$. Since $x_1, x_2 \in [0, 1)$, we must have $x_1 = x_2$. So, f is one-to-one.

(2) f is onto?

Let $y \in [0, 1)$ and let $x = \sqrt{y}$. Then $x \in [0, 1)$ and $f(x) = y$. Thus f is onto.

Therefore, f is a one-to-one correspondence, and hence it is invertible. The inverse function of f is $f^{-1} : [0, 1) \to [0, 1)$ be given by $f(x) = \sqrt{x}$. □

2.3.4 Composition of Functions

Definition 2.3.32. (Composition of functions)
Let $f : B \to C$ and $g : A \to B$ be two functions. The composition of the two functions f and g is the function $f \circ g : A \to C$ defined by $(g \circ f)(x) = g(f(x))$.

Example 2.3.33. Let $f : \mathbb{R} \to \mathbb{R}$ and let $g : \mathbb{R} \to \mathbb{R}$ be two functions given by $f(x) = x^2 + 2$ and $g(x) = \sqrt{x^2 + 4}$. Find $(g \circ f)(x)$ and $(f \circ g)(x)$.

Proof. $(g \circ f)(x) = g(f(x)) = g(x^2 + 2) = \sqrt{(x^2 + 2)^2 + 4} = \sqrt{x^4 + 4x^2 + 8}$
$(f \circ g)(x) = f(g(x)) = f(\sqrt{x^2 + 4}) = (\sqrt{x^2 + 4})^2 + 2 = x^2 + 6$ □

Remarks 2.3.34. (a) Function composition is **NOT** commutative; that is, in general, $f \circ g \neq g \circ f$.

(b) The composition of a function f and its inverse f^{-1}, in either order, yields an identity function. That is, for any x, we have

$$(f \circ f^{-1})(x) = x = (f^{-1} \circ f)(x).$$

2.3.5 Some Important Functions

Next, we introduce two important functions in discrete mathematics, namely, the floor and ceiling functions. Let x be a real number. The floor function rounds x down to the closest integer less than or equal to x, and the ceiling function rounds x up to the closest integer greater than or equal to x. These functions are often used when objects are counted. They play an important role in the analysis of the number of steps used by procedures to solve problems of a particular size.

Definition 2.3.35. (The floor and ceiling functions)

(a) The floor function assigns to the real number x the largest integer that is less than or equal to x. The value of the floor function at x is denoted by $\lfloor x \rfloor$.

(b) The ceiling function assigns to the real number x the smallest integer that is greater than or equal to x. The value of the ceiling function at x is denoted by $f(x) = \lceil x \rceil$.

Remarks 2.3.36.

(1) The floor function is often also called the greatest integer function.

(2) The ceiling function is often also called the least integer function.

Example 2.3.37. These are some values of the floor function: $\lfloor -3 \rfloor = -3$, $\lfloor 3.5 \rfloor = 3$, $\lfloor -3.5 \rfloor = -4$

Example 2.3.38. These are some values of the ceiling function: $\lceil 4 \rceil = 4$, $\lceil 3.5 \rceil = 4$, $\lceil -3.5 \rceil = -3$

The floor and ceiling functions are useful in a wide variety of applications, including those involving data storage and data transmission.

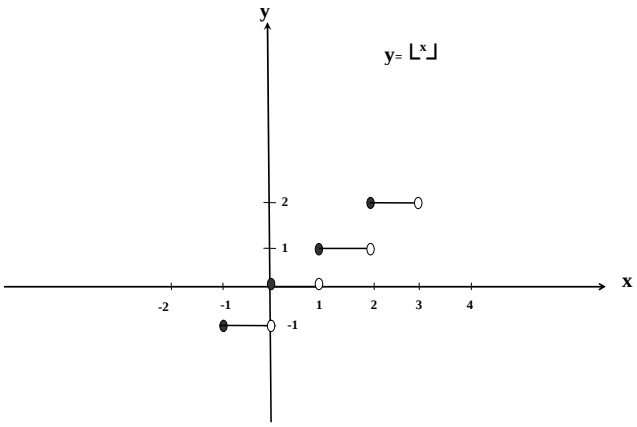


Figure 2.1: The floor function

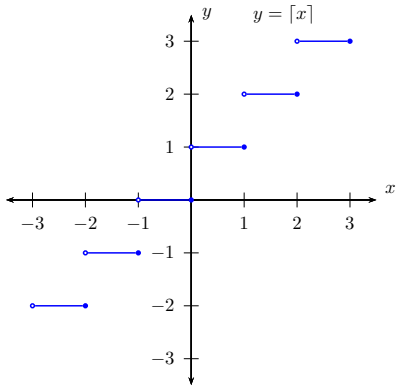


Figure 2.2: The ceiling function

Theorem 2.3.39. *(Properties of the floor and ceiling functions)*
Let n be an integer and let x be a real number.

(a) $\lfloor x \rfloor = n \quad$ *if and only if* $\quad n \leq x < n+1.$

(b) $\lceil x \rceil = n \quad$ *if and only if* $\quad n-1 < x \leq n.$

(c) $\lfloor x \rfloor = n \quad$ *if and only if* $\quad x-1 < n \leq x.$

(d) $\lceil x \rceil = n \quad$ *if and only if* $\quad x \leq n < x+1.$

(e) $x - 1 < \lfloor x \rfloor \leq x \leq \lceil x \rceil < x + 1.$

(f) $\lceil -x \rceil = -\lfloor x \rfloor$ *and* $\lfloor -x \rfloor = -\lceil x \rceil.$

(g) $\lceil x + n \rceil = \lceil x \rceil + n$ *and* $\lfloor x + n \rfloor = \lfloor x \rfloor + n.$

Proof. Properties (a), (b), (c), and (d) follow directly from the definitions. For example, (a) states that $\lfloor x \rfloor = n \quad$ if and only if $\quad n \leq x < n+1$. This is precisely what it means for

n to be the greatest integer not exceeding x, which is the definition of $\lfloor x \rfloor = n$. Properties (b), (c), and (d) can be established similarly. We will prove property (g) using a direct proof. Suppose that $\lfloor x \rfloor = m$, where m is an integer. By property (a), it follows that $m \leq x < m+1$. Adding n to all three quantities in this chain of two inequalities shows that $m+n \leq x+n < m+n+1$. Using property (a) again, we see that $\lfloor x+n \rfloor = m+n = \lfloor x \rfloor + n$. This completes the proof. Proofs of the other properties are left as exercises. □

Exercises

Page 152: 1-29, 36-39, 46-54, 73-76

2.4 Sequences and Summations

Sequences are ordered lists of elements, used in discrete mathematics in many ways. For example, they can be used to represent solutions to certain counting problems, as we will see in Chapter 8. They are also an important data structure in computer science. We will often need to work with sums of terms of sequences in our study of discrete mathematics. This section reviews the use of summation notation, basic properties of summations, and formulas for the sums of terms of some particular types of sequences.

The terms of a sequence can be specified by providing a formula for each term of the sequence. In this section we describe another way to specify the terms of a sequence using a recurrence relation, which expresses each term as a combination of the previous terms. We will introduce one method, known as iteration, for finding a closed formula for the terms of a sequence specified via a recurrence relation. Identifying a sequence when the first few terms are provided is a useful skill when solving problems in discrete mathematics. We will provide some tips, including a useful tool on the Web, for doing so.

2.4.1 Sequences

A sequence is a discrete structure used to represent an ordered list. For example, $1, 2, 3, 5, 8$ is a sequence with five terms and $1, 3, 9, 27, 81, \ldots, 3^n, \cdots$ is an infinite sequence.

Definition 2.4.1.

(1) A sequence is a function from a subset of the set of integers (usually either the set $\{0, 1, 2, \cdots\}$ or the set $\{1, 2, 3, \cdots\}$) to a set S.

(2) We use the notation a_n to denote the image of the integer n. We call a_n a term of the sequence.

Notation 2.4.2. We use the notation $\{a_n\}$ to describe the sequence.

Remark 2.4.3. Note that a_n represents an individual term of the sequence $\{a_n\}$. Be aware that the notation $\{a_n\}$ for a sequence conflicts with the notation for a set. However, the context in which we use this notation will always make it clear when we are dealing with sets and when we are dealing with sequences.

Example 2.4.4. Let $a_n = -4n$, Then $\{a_n\}$ is a sequence. The list of the terms of this sequence starts with

$$-4, -8, -12, \cdots$$

Example 2.4.5. If $a_n = 3$, then $\{a_n\}$ is a sequence with all terms are equal 3.

Example 2.4.6. $\{a_n\} = \{(-1)^n\}$ is a sequence with terms $\{-1, 1, -1, \cdots\}$.

Definition 2.4.7. (Geometric progression)
A geometric progression is a sequence of the form

$$a, ar, ar^2, \ldots, ar^n, \ldots,$$

where the initial term a and the common ratio r are real numbers.

Example 2.4.8. The sequence $\{a_n\} = \{2 \cdot (-1)^n\}$ is a geometric progression with initial term $a = 2$ and common ratio $r = -1$. The list of terms begins with

$$2, -2, 2, -2, \ldots$$

Definition 2.4.9. (Arithmetic progression)
An arithmetic progression is a sequence of the form

$$a, a + d, a + 2d, \ldots, a + nd, \ldots,$$

where the initial term a and the common difference d are real numbers.

Example 2.4.10. The sequence $\{b_n\}$ with $b_n = -2 + 3n$ is an arithmetic progression with initial term $a = -2$ and common difference $d = 3$. The list of terms begins with

$$-2, 1, 4, 7, \ldots$$

Sequences of the form $a_1, a_2, \ldots, a_n$ are often used in computer science. These finite sequences are also called strings. This string is also denoted by $a_1a_2 \ldots a_n$. (Recall that bit strings, which are finite sequences of bits, were introduced in Section 1.1.) The length of a string is the number of terms in this string. The empty string, denoted by λ, is the string that has no terms. The empty string has length zero.

Example 2.4.11. The string $abcdef$ is string of length 6.

2.4.2 Recurrence Relations

Sequences may be defined by given a_n directly. Alternatively sequences may be defined by a rule for calculating any later term from terms that precede it and by giving the necessary initial terms.

Definition 2.4.12. (A recurrence relation)

(1) A recurrence relation for the sequence $\{a_n\}$ is an equation that expresses a_n in terms of one or more of the previous terms of the sequence, namely, $a_0, a_1, \ldots, a_{n-1}$, for all integers n with $n \geq n_0$, where n_0 is a nonnegative integer.

(2) A sequence is called a solution of a recurrence relation if its terms satisfy the recurrence relation. (A recurrence relation is said to recursively define a sequence.)

Example 2.4.13. Let $a_1 = 2, \quad a_{n+1} = n - a_n$. Find the first 5 terms of this sequence.

Proof. $a_1 = 2,\ a_2 = -1,\ a_3 = 3,\ a_4 = 0,\ a_5 = 4$ □

Example 2.4.14. Find the first 5 terms of the sequence $a_1 = 1,\ a_2 = 2, \quad a_{n+2} = 2a_n + a_{n+1}$.

Proof. $a_1 = 1,\ a_2 = 2,\ a_3 = 4,\ a_4 = 8,\ a_5 = 16$ □

Next, we define a particularly useful sequence defined by a recurrence relation, known as the Fibonacci sequence, after the Italian mathematician Fibonacci who was born in the 12th century. It is important for many applications, including modeling the population growth of rabbits.

Definition 2.4.15. (The Fibonacci sequence)
The Fibonacci sequence, $f_0, f_1, f_2, \ldots$, is defined by the initial conditions $f_0 = 0,\ f_1 = 1$, and the recurrence relation $f_n = f_{n-1} + f_{n-2}$ for $n = 2, 3, 4, \ldots$.

Example 2.4.16. Find the Fibonacci numbers f_2, f_3, f_4, f_5, and f_6.

Proof. Using the recurrence relation for the Fibonacci sequence we find that

$$f_2 = f_1 + f_0 = 1, f_3 = 2, f_4 = 3, f_5 = 5, f_6 = 8.$$

□

Example 2.4.17. Consider the sequence of factorials $\{a_n\}$ defined by $a_n = n!, \quad n \geq 0$. By definition $n! = n(n-1)!$ and hence it satisfies the recurrence relation $a_n = na_{n-1}$, together with the initial condition $a_0 = 1$.

Example 2.4.18. Determine whether the sequence $\{a_n\}$, where $a_n = 3n$ for every nonnegative integer n, is a solution of the recurrence relation $a_n = 2a_{n-1} - a_{n-2}$ for $n = 2, 3, 4, \ldots$.

Proof. Let $a_n = 3n$. Then $2a_{n-1} - a_{n-2} = 2(3(n-1)) - 3(n-2) = 3n = a_n$. Therefore, $\{a_n\}$ is a solution of the recurrence relation. □

Example 2.4.19. Determine whether the sequence $\{a_n\}$, where $a_n = n^2$ for every nonnegative integer n, is a solution of the recurrence relation $a_n = a_{n-1} + a_{n-2}$ for $n = 2, 3, 4, \ldots$.

Proof. $a_{n-1} + a_{n-2} = (n-1)^2 + (n-2)^2 \neq a_n$. □

Solving Recurrence Relations

There are many methods for solving recurrence relations. Here, we will introduce a straightforward method known as iteration via several examples.

Example 2.4.20. Solve the recurrence relation $a_n = a_{n-1} + 3$ for $n = 1, 2, 3, \ldots$ with the initial condition $a_0 = 2$.

Proof. We can successively apply the recurrence relation starting with the initial condition $a_0 = 2$, and working upward until we reach a_n to deduce a closed formula for the sequence. We see that

$$\begin{aligned}
a_1 &= a_0 + 3 = 2 + 3 \\
a_2 &= a_1 + 3 = (2 + 3) + 3 = 2 + 3 \cdot 2 \\
a_3 &= a_2 + 3 = (2 + 2 \cdot 3) + 3 = 2 + 3 \cdot 3 \\
&\vdots \\
a_n &= a_{n-1} + 3 = (2 + 3 \cdot (n-1)) + 3 = 2 + 3n.
\end{aligned}$$

We can also successively apply the recurrence relation starting with the term a_n and working downward until we reach the initial condition $a_0 = 2$ to deduce this same formula. The steps are

$$\begin{aligned}
a_n &= a_{n-1} + 3 \\
&= (a_{n-2} + 3) + 3 = a_{n-2} + 3 \cdot 2 \\
&= (a_{n-3} + 3) + 3 \cdot 2 = a_{n-3} + 3 \cdot 3 \\
&\vdots \\
&= a_0 + 3n = 2 + 3n.
\end{aligned}$$

Note that this sequence is an arithmetic progression. □

Example 2.4.21. Solve the recurrence relation $a_n = 4na_{n-1}$ for $n = 1, 2, 3, \ldots$ with the initial condition $a_0 = 5$.

Proof. We can successively apply the recurrence relation starting with the initial condition $a_0 = 5$, and working upward until we reach a_n to deduce a closed formula for the sequence. We see that

$$\begin{aligned}
a_1 &= 4a_0 \\
a_2 &= 4 \cdot 2a_1 = 4^2 \cdot 2a_0 \\
a_3 &= 4 \cdot 3a_3 = 4^3 \cdot 2 \cdot 3a_0 \\
&\vdots \\
a_n &= 5 \cdot 4^n \cdot n!.
\end{aligned}$$

□

The technique used in the above examples is called iteration. We have iterated, or repeatedly used, the recurrence relation. The first approach is called forward substitution- we found successive terms beginning with the initial condition and ending with a_n. The second approach is called backward substitution, because we began with a_n and iterated to express it in terms of falling terms of the sequence until we found it in terms of a_0. Note that when we use iteration, we essential guess a formula for the terms of the sequence. To prove that our guess is correct, we need to use mathematical induction, a technique we discuss in Chapter 5.

The recurrence relations can be used to model a wide variety of problems such as compound interest.

Special Integer Sequences

A common problem in discrete mathematics is finding a closed formula, a recurrence relation, or some other type of general rule for constructing the terms of a sequence. Sometimes only a few terms of a sequence solving a problem are known; the goal is to identify the sequence. Even though the initial terms of a sequence do not determine the entire sequence (after all, there are infinitely many different sequences that start with any finite set of initial terms), knowing the first few terms may help you make an educated conjecture about the identity of your sequence. Once you have made this conjecture, you can try to verify that you have the correct sequence.

When trying to deduce a possible formula, recurrence relation, or some other type of rule for the terms of a sequence when given the initial terms, try to find a pattern in these terms. You might also see whether you can determine how a term might have been produced from those preceding it. There are many questions you could ask, but some of the more useful are:

- Are there runs of the same value? That is, does the same value occur many times in a row?
- Are terms obtained from previous terms by adding the same amount or an amount that depends on the position in the sequence?
- Are terms obtained from previous terms by multiplying by a particular amount?
- Are terms obtained by combining previous terms in a certain way?
- Are there cycles among the terms?

Example 2.4.22. Find formulae for the sequences with the following first five terms: $1, 4, 9, 16, 25, \cdots$.

Proof. We recognize that the terms are the square of n. The sequence with $a_n = n^2$, $n = 1, 2, 3, \cdots$ is a possible match. □

Example 2.4.23. Find formulae for the sequences with the following first five terms: $1, 4, 7, 10, 13, \cdots$.

Proof. We note that each term is obtained by adding 3 to the previous term. The sequence with $a_n = 3n+1$, $n = 0, 1, 2, \cdots$ is a possible match. This proposed sequence is an arithmetic progression with $a = 1$ and $d = 3$. □

Example 2.4.24. Find formulae for the sequences with the following first five terms: $-1, \frac{1}{2}, \frac{-1}{4}, \frac{1}{8}, \frac{-1}{16}, \cdots$.

Proof. We note that the nominators alternate between 1 and -1 and the denominators are powers of 2. The sequence with $a_n = \frac{(-1)^{n+1}}{2^n}$, $n = 0, 1, 2, \cdots$ is a possible match. This proposed sequence is a a geometric progression with $a = -1$ and $r = \frac{-1}{2}$. □

Example 2.4.25. How can we produce the terms of a sequence if the first 10 terms are $5, 11, 17, 23, 29, 35, 41, 47, 53, 59$?

Proof. Note that each of the first 10 terms of this sequence after the first is obtained by adding 6 to the previous term. (We could see this by noticing that the difference between consecutive terms is 6.) Consequently, the nth term could be produced by starting with 5 and adding 6 a total of $n-1$ times; that is, a reasonable guess is that the nth term is $a_n = 5 + 6(n-1) = 6n - 1$, $n = 1, 2, 3, \cdots$. (This is an arithmetic progression with $a = 5$ and $d = 6$.) □

Example 2.4.26. How can we produce the terms of a sequence if the first 10 terms are $1, 3, 4, 7, 11, 18, 29, 47, 76, 123$?

Proof. Observe that each successive term of this sequence, starting with the third term, is the sum of the two previous terms. Consequently, if a_n is the nth term of this sequence, we guess that the sequence is determined by the recurrence relation $a_n = a_{n-1} + a_{n-2}$ with initial conditions $a_1 = 1$ and $a_2 = 3$. The same recurrence relation as the Fibonacci sequence, but with different initial conditions. This sequence is known as the Lucas sequence, after the French mathematician François Édouard Lucas. Lucas studied this sequence and the Fibonacci sequence in the nineteenth century. □

Remark 2.4.27. Another useful technique for finding a rule for generating the terms of a sequence is to compare the terms of a sequence of interest with the terms of a well-known integer sequence, such as terms of an arithmetic progression, terms of a geometric progression, perfect squares, perfect cubes, and so on.

Summations

Next, we consider the addition of the terms of a sequence. For this we introduce summation notation. We begin by describing the notation used to express the sum of the terms

$$a_m, a_{m+1}, \cdots, a_n$$

from the sequence $\{a_n\}$.

Definition 2.4.28. (Sigma notation for finite sums)
The $\sum_{k=m}^{n} a_k = \sum_{m \le k \le n} a_k$ (read: the sum from $k = m$ to $k = n$ of a_k) denotes the sum $a_m + a_{m+1} + \cdots + a_n$.

- The a_k are the terms of the sum.
- The variable k is the index of summation.
- m is the lower limit of summation.
- n is the upper limit of summation.

Example 2.4.29. $\sum_{k=1}^{4} k^2 = 1 + 4 + 9 + 16 = 30$.

Example 2.4.30. $\sum_{k=3}^{7}(-1)^k = -1+1-1+1-1 = -1.$

Example 2.4.31. Write $2+4+6+8+10$ in sigma notation.

Proof. □

Example 2.4.32. Write $\frac{1}{7}+\frac{2}{7}+\frac{3}{7}+\frac{4}{7}+\frac{5}{7}+\frac{6}{7}$ in sigma notation.

Proof. □

2.4.3 Algebra Rules for Finite Sums

The usual laws for arithmetic apply to summations. Thus

(1) $\sum_{k=1}^{n}(a_k \pm b_k) = \sum_{k=1}^{n} a_k \pm \sum_{k=1}^{n} b_k.$

(2) $\sum_{k=1}^{n} ca_k = c\sum_{k=1}^{n} a_k$, where c is any number.

(3) $\sum_{k=1}^{n} c = nc$, where c is any number.

Example 2.4.33. Find $\sum_{k=1}^{9} 3.$

Proof. □

Example 2.4.34. If $\sum_{k=1}^{30} a_k = 7$ and $\sum_{k=1}^{30} b_k = -4$, then find $\sum_{k=1}^{30}(a_k + 3b_k)$.

Proof. □

2.4.4 Reindexing (Changing indexes)

Sometimes it is useful to shift the index of summation in a sum. This is often done when two sums need to be added but their indices of summation do not match. As long as we preserve the order of the terms of a summation, we can change the index of the summation without altering its value.

Example 2.4.35. Write $\sum_{j=5}^{n} \frac{3}{9^j}$ in the form $\sum_{k=1}^{m} a_k$.

Proof. $\sum_{j=5}^{n} \frac{3}{9^j} \overset{k=j-4}{=\!=\!=} \sum_{k=1}^{n-4} \frac{3}{9^{k+4}}$ □

Geometric series

Sums of terms of geometric progressions are called geometric series. They commonly arise and the following theorem gives us a formula for the sum of terms of a geometric progression.

Theorem 2.4.36. *If a and r are real numbers and $r \neq 0$, then* $\sum_{k=0}^{n} ar^k = \begin{cases} \dfrac{a(r^{n+1}-1)}{r-1}, & r \neq 1 \\ a(n+1), & r = 1. \end{cases}$

Proof. □

Example 2.4.37. Find the value of the sum $\sum_{j=0}^{6}[6 \cdot (-1)^j - (2)^j]$

Proof. $\sum_{j=0}^{6}[6 \cdot (-1)^j - (2)^j] = \sum_{j=0}^{6} 6 \cdot (-1)^j - \sum_{j=0}^{6}(2)^j = \dfrac{6[(-1)^7 - 1]}{-1-1} - \dfrac{(2^7-1)}{2-1} = 6 - 127 = -121$ □

Double summations

Double summations arise in many contexts (as in the analysis of nested loops in computer programs). To evaluate the double sum, first expand the inner summation and then continue by computing the outer summation.

Example 2.4.38. Find the value of $\sum_{j=1}^{4}\sum_{k=1}^{2}(j+k)$.

Proof. $\sum_{j=1}^{4}\sum_{k=1}^{2}(j+k) = \sum_{j=1}^{4}[(j+1)+(j+2)] = (2+3)+(3+4)+(4+5)+(5+6) = 32.$ □

Some Useful Summation Formulae

(1) $\sum_{k=1}^{n} k = \dfrac{n(n+1)}{2}$.

(2) $\sum_{k=1}^{n} k^2 = \dfrac{n(n+1)(2n+1)}{6}$.

(3) $\sum_{k=1}^{n} k^3 = \left[\dfrac{n(n+1)}{2}\right]^2$.

Example 2.4.39. Find $\sum_{k=1}^{6} k$.

Proof. 21 □

Example 2.4.40. Find $\sum_{k=1}^{14}(k^3+3k)$.

Proof. $\sum_{k=1}^{14}(k^3+3k) = \sum_{k=1}^{14}k^3 + 3\sum_{k=1}^{14}k = \left(\frac{14(15)}{2}\right)^2 + 3\cdot\frac{14(15)}{2} = 11340$ □

Example 2.4.41. Find $\sum_{k=5}^{20}k^2$.

Proof. $\sum_{k=5}^{20}k^2 = \sum_{k=1}^{20}k^2 - \sum_{k=1}^{4}k^2 = \frac{20(21)(41)}{6} - \frac{4(5)(9)}{6}$ □

Exercises

Page 167: 1-17, 25, 29, 31-37, 39, 40

2.5 Omit

2.6 Matrices

Matrices are used throughout discrete mathematics to express relationships between elements in sets. In subsequent chapters we will use matrices in a wide variety of models. For instance, matrices will be used in models of communications networks and transportation systems. Many algorithms will be developed that use these matrix models. This section reviews matrix arithmetic that will be used in these algorithms.

Definition 2.6.1. A matrix is a rectangular array of numbers or other mathematical objects arranged in rows and columns. It has the general form

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}.$$

In a matrix the mathematical object and its position in the array are important.

The size of a matrix is defined by the number of rows and columns that it contains. A matrix with m rows and n columns is called an $m \times n$ matrix or $m-$by$-n$ matrix, while m and n are called its dimensions. The individual items in a matrix are called its elements or entries. A square matrix is an $n \times n$ matrix.

Example 2.6.2. The matrix $A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$ is a 2×3 matrix.

A convenient shorthand notation for expressing the matrix A is to write $A = [a_{ij}]$, which indicates that A is the matrix with its (i, j)th element equal to a_{ij}.

2.6.1 Matrix Arithmetic

Now we will discuss the basic operations of matrix arithmetic.

Definition 2.6.3. (Matrix addition)
Let $A = [a_{ij}]$ and $B = [b_{ij}]$ be $m \times n$ matrices. The sum of A and B, denoted by $A + B$, is the $m \times n$ matrix $A + B = [a_{ij} + b_{ij}]$.

Remark 2.6.4. The sum of two matrices of the same size is obtained by adding elements in the corresponding positions. Matrices of different sizes cannot be added, because the sum of two matrices is defined only when both matrices have the same number of rows and the same number of columns.

Example 2.6.5. Find $A + B$ if $A = \begin{bmatrix} 1 & 2 \\ 4 & 5 \\ 7 & 8 \end{bmatrix}$ and $B = \begin{bmatrix} 2 & 3 \\ 8 & 10 \\ 7 & 9 \end{bmatrix}$.

Proof. A and B are 3×2 matrices. Thus $A + B$ is defined and

$$A + B = \begin{bmatrix} 1 & 2 \\ 4 & 5 \\ 7 & 8 \end{bmatrix} + \begin{bmatrix} 2 & 3 \\ 8 & 10 \\ 7 & 9 \end{bmatrix} = \begin{bmatrix} 3 & 5 \\ 12 & 15 \\ 14 & 17 \end{bmatrix}.$$

□

Definition 2.6.6. (Matrix product)
Let A be an $m \times k$ matrix and B be a $k \times n$ matrix. The product of A and B, denoted by AB, is the $m \times n$ matrix with its (i, j)th entry equal to the sum of the products of the corresponding elements from the ith row of A and the jth column of B. In other words, if $A = [a_{ij}]$, $B = [b_{ij}]$ and $AB = [c_{ij}]$, then

$$c_{ij} = a_{i1}b_{1j}a_{i2}b_{2j} + \cdots + a_{ik}b_{kj}.$$

Remark 2.6.7. The product of two matrices is not defined when the number of columns in the first matrix and the number of rows in the second matrix are not the same.

Example 2.6.8. Let $A = \begin{bmatrix} 2 & 2 & -2 \\ 3 & 2 & 4 \\ 0 & 2 & -1 \end{bmatrix}$ and $B = \begin{bmatrix} 2 & 3 \\ 8 & 10 \\ 7 & 9 \end{bmatrix}$. Find AB if it is defined.

Proof. A is a 3×3 matrix and B is 3×2 matrix. Thus AB is defined and it is 3×2 matrix.

$$AB = \begin{bmatrix} 6 & 8 \\ 50 & 65 \\ 9 & 11 \end{bmatrix}.$$

□

Example 2.6.9. Let $A = \begin{bmatrix} 3 & 2 \\ 2 & -4 \\ 3 & -2 \end{bmatrix}$ and $B = \begin{bmatrix} 2 & 3 & -1 & 6 \\ 8 & 3 & 7 & 9 \end{bmatrix}$. Find AB.

Proof. A is a 3×2 matrix and B is 2×4 matrix. Thus AB is defined and it is 3×4 matrix.

$$AB = \begin{bmatrix} 22 & 15 & 11 & 36 \\ -28 & -6 & -30 & -24 \\ -10 & 3 & -17 & 0 \end{bmatrix}.$$

□

Remark 2.6.10. Matrix multiplication is not commutative. That is, if A and B are two matrices, then in general $AB \neq BA$.

Transposes and Powers of Matrices

We now introduce an important matrix with entries that are zeros and ones.

Definition 2.6.11. (Identity matrix)
The identity matrix of order n is the $n \times n$ matrix $I_n = [d_{ij}]$, where $d_{ij} = 1$ if $i = j$ and $d_{ij} = 0$ if $i \neq j$. Hence

$$I_n = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}.$$

Remark 2.6.12. If A is an $m \times n$ matrix, then $AI_n = I_m A = A$.

Example 2.6.13. If $A = \begin{bmatrix} 2 & 2 & 6 \\ 1 & 5 & 4 \end{bmatrix}$, then $AI_3 = \begin{bmatrix} 2 & 2 & 6 \\ 1 & 5 & 4 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = A$ and $I_2 A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 2 & 6 \\ 1 & 5 & 4 \end{bmatrix} = A$.

Definition 2.6.14. (Powers of square matrix)
If A is an $n \times n$ square matrix, then we define

$$A^0 = I_n, \quad A^2 = AA, \quad A^r = AA^{r-1}.$$

Example 2.6.15. Find A^3 if $A = \begin{bmatrix} 1 & 2 \\ 3 & 2 \end{bmatrix}$.

Proof.

$$A^2 = \begin{bmatrix} 1 & 2 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 2 \end{bmatrix} = \begin{bmatrix} 7 & 6 \\ 9 & 10 \end{bmatrix}$$

$$A^3 = \begin{bmatrix} 1 & 2 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} 7 & 6 \\ 9 & 10 \end{bmatrix} = \begin{bmatrix} 25 & 26 \\ 39 & 38 \end{bmatrix}$$

□

Definition 2.6.16. (Matrix Transpose)
Let $A = [a_{ij}]$ be an $m \times n$ matrix. The transpose of A, denoted by A^t, is the $n \times m$ matrix obtained by interchanging the rows and columns of A. In other words, if $A^t = [b_{ij}]$, then $b_{ij} = a_{ji}$ for $i = 1, 2, \cdots, n$ and $j = 1, 2, \cdots, m$.

Example 2.6.17. Find A^t if $A = \begin{bmatrix} 2 & 2 & -2 & 2 \\ 0 & 2 & 4 & 2 \\ 6 & 0 & -1 & 4 \end{bmatrix}$.

Proof. □

Symmetric matrix

Definition 2.6.18. (Symmetric matrix)
A square matrix A is called symmetric if $A = A^t$.

Remark 2.6.19. If $A = [a_{ij}]$ is symmetric if $a_{ij} = a_{ji}$ for all i and j with $1 \leq i \leq n$ and $1 \leq j \leq n$.

Example 2.6.20. The matrix $A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 5 & 6 \\ 3 & 6 & 9 \end{bmatrix}$ is symmetric.

Zero-One Matrices

A matrix all of whose entries are either 0 or 1 is called a zero-one matrix. Zero-one matrices are often used to represent discrete structures, as we can see in Chapters 9 and 10. Algorithms using these structures are based on Boolean arithmetic with zero-one matrices. This arithmetic is based on the Boolean operations $\wedge$ and $\vee$ operating on pairs of bits.

Definition 2.6.21. (Join and Meet)
Let $A = [a_{ij}]$ and $B = [b_{ij}]$ be $m \times n$ zero-one matrices.

(1) The join of A and B, denoted by $A \vee B$, is the zero-one matrix with (i, j)th entry $a_{ij} \vee b_{ij}$.

(2) The meet of A and B, denoted by $A \wedge B$, is the zero-one matrix with (i, j)th entry $a_{ij} \wedge b_{ij}$.

Example 2.6.22. Find the join and meet of A and B if $A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 0 \end{bmatrix}$ and $B = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$.

Proof. The join of A and B is

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

The meet of A and B is

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

□

Definition 2.6.23. (Boolean product)
Let $A = [a_{ij}]$ be an $m \times k$ zero-one matrix and $B = [b_{ij}]$ be a $k \times n$ zero-one matrix. Then the Boolean product of A and B, denoted by $A \odot B$, is the $m \times n$ matrix with (i, j)th entry c_{ij} where

$$c_{ij} = (a_{i1} \wedge b_{1j}) \vee (a_{i2} \wedge b_{2j}) \vee \cdots (a_{ik} \wedge b_{kj}).$$

Note that the Boolean product of A and B is obtained in an analogous way to the ordinary product of these matrices, but with addition replaced with the operation $\vee$ and with multiplication replaced with the operation $\wedge$.

Example 2.6.24. Find the Boolean product of A and B if $A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 0 \end{bmatrix}$ and $B = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$.

Proof. The Boolean product of A and B is

$$A \odot B = \begin{bmatrix} (1 \wedge 0) \vee (0 \wedge 0) & (1 \wedge 0) \vee (0 \wedge 1) \\ (1 \wedge 0) \vee (1 \wedge 0) & (1 \wedge 0) \vee (1 \wedge 1) \\ (0 \wedge 0) \vee (0 \wedge 0) & (0 \wedge 0) \vee (0 \wedge 1) \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$$

□

We can also define the Boolean powers of a square zero-one matrix. These powers will be used in our subsequent studies of paths in graphs, which are used to model such things as communications paths in computer networks.

Definition 2.6.25. (Boolean powers of square matrix)
Let A be an $n \times n$ square zero-one matrix and let r be a positive integer. The rth Boolean power of A, denoted by $A^{[r]}$, is the Boolean product of r factors of A

$$A^{[r]} = \underbrace{A \odot A \odot \cdots \odot A}_{r \text{ times}}.$$

We also define $A^{[0]} = I_n$.

Example 2.6.26. Find $A^{[n]}$ for all positive integers n if $A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$.

Proof.

$$A^{[2]} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \odot \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} (1 \wedge 1) \vee (0 \wedge 1) & (1 \wedge 0) \vee (0 \wedge 1) \\ (1 \wedge 1) \vee (1 \wedge 1) & (1 \wedge 0) \vee (1 \wedge 1) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} = A.$$

Therefore, $A^{[n]} = A$ for any positive integers n. □

Example 2.6.27. Let A, B, and C be $m \times n$ zero-one matrices. Show that $A \vee (B \vee C) = (A \vee B) \vee C$.

Proof. Let $A = [a_{ij}]$, $B = [b_{ij}]$, $C = [c_{ij}]$, $(A \vee B) \vee C = [d_{ij}]$, and $A \vee (B \vee C) = [e_{ij}]$. Then

$$d_{ij} = (a_{ij} \vee b_{ij}) \vee c_{ij} = a_{ij} \vee (b_{ij} \vee c_{ij}) = e_{ij}.$$

□

Exercises

Page 183: 1-11,18-20, 26-33

Chapter 3

Algorithms

3.1 Omit

3.2 The Growth of Functions

The time required to solve a problem depends on the number of operations it uses, the hardware, and the software used to run the program that implements the algorithm. However, when we change the hardware and software used to implement an algorithm, we can closely approximate the time required to solve a problem of size n by multiplying the previous time required by a constant.

Big-O notation is used extensively to estimate the number of operations an algorithm uses as its input grows. With the help of this notation, we can determine whether it is practical to use a particular algorithm to solve a problem as the size of the input increases. Furthermore, using big-O notation, we can compare two algorithms to determine which is more efficient as the size of the input grows.

This section introduces big-O notation and the related big-Omega and big-Theta notations. We will explain how big-O, big-Omega, and big-Theta estimates are constructed and establish estimates for some important functions that are used in the analysis of algorithms.

3.2.1 Big-O Notation

The growth of functions is often described using a special notation.

Definition 3.2.1. Let f and g be functions from the set of integers or the set of real numbers to the set of real numbers. We say that $f(x)$ is $O(g(x))$ [read as "$f(x)$ is big-oh of $g(x)$"] if there are constants C and k such that

$$|f(x)| \leq C|g(x)|$$

whenever $x > k$.

Remarks 3.2.2.

(1) Intuitively, the definition that $f(x)$ is $O(g(x))$ says that $f(x)$ grows slower that some fixed multiple of $g(x)$ as x grows without bound.

(2) The constants C and k in the definition of big-O notation are called witnesses to the relationship $f(x)$ is $O(g(x))$.

(3) To establish that $f(x)$ is $O(g(x))$ we need only one pair of witnesses to this relationship. That is, to show that $f(x)$ is $O(g(x))$, we need find only one pair of constants C and k, the witnesses, such that $|f(x)| \leq C|g(x)|$ whenever $x > k$.

(4) If there is one pair of witnesses to the relationship $f(x)$ is $O(g(x))$, there are infinitely many pairs of witnesses.

Example 3.2.3. Show that $f(x) = x^2 + x + 3$ is $O(x^2)$.

Proof. A useful approach for finding a pair of witnesses is to first select a value of k for which the size of $|f(x)|$ can be readily estimated when $x > k$ and to see whether we can use this estimate to find a value of C for which $|f(x)| \leq C|g(x)|$ for $x > k$.

Let $k = 1$. Then $x < x^2$ and $1 < x^2$ for $x > 1$. It follows that

$$0 \leq x^2 + x + 3 \leq x^2 + x^2 + 3x^2 = 5x^2$$

whenever $x > 1$. Consequently, we can take $C = 5$ and $k = 1$ as witnesses to show that $f(x)$ is $O(x^2)$. That is, $|f(x)| = x^2 + x + 3 < 5x^2$ whenever $x > 1$. (Note that it is not necessary to use absolute values here because all functions in these equalities are positive when x is positive.) □

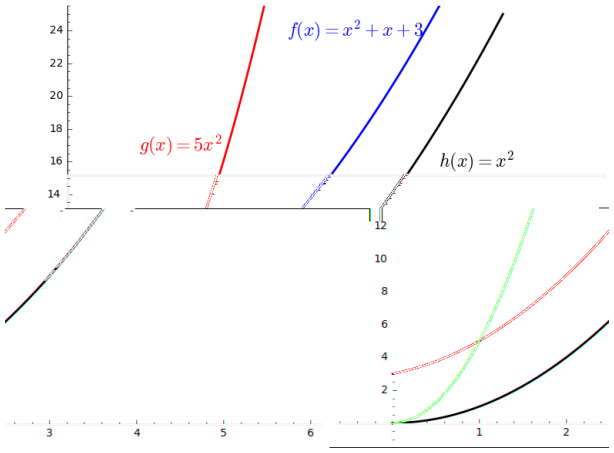


Figure 3.1: The Function $x^2 + x + 3$ is $O(x^2)$

Example 3.2.4. Show that $8x^2$ is $O(x^3)$.

Proof. Let $k = 1$. Then for $x > 1$ we have

$$x^2 \leq x^3$$

and hence

$$|f(x)| = 8x^2 \leq 8x^3 \quad \forall x > 1.$$

Consequently, we can take $C = 8$ and $k = 1$ as witnesses to establish the relationship $8x^2$ is $O(x^3)$. □

Example 3.2.5. Show that n^2 is not $O(n)$.

Proof. To show that n^2 is not $O(n)$, we must show that no pair of witnesses C and k exist such that $n^2 \leq Cn$ whenever $n > k$. We will use a proof by contradiction to show this.

Suppose that there are constants C and k for which $|n^2| \leq C|n|$ whenever $n > k$. Observe that when $n > 0$ we can divide both sides of the inequality by n to obtain the equivalent inequality $n \leq C$. However, no matter what C and k are, the inequality $n \leq C$ cannot hold for all n with $n > k$. In particular, once we set a value of k, we see that when n is larger than the maximum of k and C, it is not true that $n \leq C$ even though $n > k$. This contradiction shows that n^2 is not $O(n)$. □

Example 3.2.6. Is it true that x^3 is $O(8x^2)$?

Proof. To determine whether x^3 is $O(8x^2)$, we need to determine whether witnesses C and k exist, so that $x^3 \leq C * 8x^2$ whenever $x > k$. We will show that no such witnesses exist using a proof by contradiction.

Assume that there are witnesses C and k such that the inequality $x^3 \leq C * 8x^2$ holds for all $x > k$. Observe that the inequality $x^3 \leq C * 8x^2$ is equivalent to the inequality $x \leq 8C$, which follows by dividing both sides by the positive quantity x^2. However, no matter what C is, it is not the case that $x \leq 8C$ for all $x > k$ no matter what k is, because x can be made arbitrarily large. It follows that no witnesses C and k exist for this proposed big-O relationship. Hence, x^3 is not $O(87x^2)$. □

Remark 3.2.7. The fact that $f(x)$ is $O(g(x))$ is sometimes written $f(x) = O(g(x))$. However, the equals sign in this notation does not represent a genuine equality. Rather, this notation tells us that an inequality holds relating the values of the functions f and g for sufficiently large numbers in the domains of these functions. However, it is acceptable to write $f(x) \in O(g(x))$ because $O(g(x))$ represents the set of functions that are $O(g(x))$.

3.2.2 Big-O Estimates for Some Important Functions

Polynomials can often be used to estimate the growth of functions. Instead of analyzing the growth of polynomials each time they occur, we would like a result that can always be used to estimate the growth of a polynomial.

Theorem 3.2.8. *Let $f(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$, where $a_0, a_1, \cdots, a_{n-1}, a_n$ are real numbers. Then $f(x)$ is $O(x^n)$.*

Proof. Let $x > 1$. Then $x < x^2 < \cdots < x^n$. Using the triangle inequality we obtain

$$\begin{aligned} |f(x)| &\leq |a_n|x^n + |a_{n-1}|x^{n-1} + \cdots + |a_1|x + |a_0| \\ &\leq x^n(|a_n| + |a_{n-1}| + \cdots + |a_1| + |a_0|) = Cx^n. \end{aligned}$$

□

Example 3.2.9. Find a big-O estimate for the sum of the first n positive integers.

Proof. The sum of the first n positive integers is $f(n) = 1 + 2 + \cdots + n$. Since each of the integers in the sum does not exceed n, it follows that

$$f(n) = 1 + 2 + \cdots + n \leq n + n + \cdots + n = n^2.$$

Thus $f(n) = 1 + 2 + \cdots + n$ is $O(n^2)$, taking $C = 1$ and $k = 1$ as witnesses. □

Example 3.2.10. Consider the factorial function $f(n) = n!$ is defined by $n! = 1 \cdot 2 \cdot 3 \cdot \cdots \cdot n$.

(a) Give big-O estimates for $f(n)$.

(b) Give big-O estimates for $\log f(n)$.

Proof.

(a) A big-O estimate for $n!$ can be obtained by noting that each term in the product does not exceed n. Hence,

$$n! = 1 \cdot 2 \cdot 3 \cdot \cdots \cdot n \leq n \cdot n \cdot n \cdot \cdots \cdot n = n^n.$$

This inequality shows that $n!$ is $O(n^n)$, taking $C = 1$ and $k = 1$ as witnesses.

(b) Taking logarithms of both sides of the inequality established for $n!$, we obtain

$$\log n! \leq \log n^n = n \log n.$$

This implies that $\log n!$ is $O(n \log n)$, again taking $C = 1$ and $k = 1$ as witnesses.

□

3.2.3 The Growth of Combinations of Functions

Many algorithms are made up of two or more separate subprocedures. The number of steps used by a computer to solve a problem with input of a specified size using such an algorithm is the sum of the number of steps used by these subprocedures. To give a big-O estimate for the number of steps needed, it is necessary to find big=O estimates for the number of steps used by each subprocedure and then combine these estimates.

Theorem 3.2.11. *Suppose that $f_1(x)$ is $O(g_1(x))$ and that $f_2(x)$ is $O(g_2(x))$. Then $(f_1 + f_2)(x)$ is $O(\max(|g_1(x)|, |g_2(x)|))$.*

Proof. Suppose that $f_1(x)$ is $O(g_1(x))$ and that $f_2(x)$ is $O(g_2(x))$. Then $\exists C_1,\ \exists C_2,\ \exists k_1,\ \exists k_2$ such that

$$|f_1(x)| \leq C_1|g_1(x)| \quad \text{for } x > k_1$$

and

$$|f_2(x)| \leq C_2|g_2(x)| \quad \text{for } x > k_2.$$

Let $g(x) = \max(|g_1(x)|, |g_2(x)|)$ and let $k = \max(k_1, k_2)$. Then

$$|f_1(x) + f_2(x)| \leq |f_1(x)| + |f_2(x)| \leq C_1|g_1(x)| + C_2|g_2(x)| \leq (C_1 + C_2)|g(x)| \qquad \text{for } x > k.$$

□

Corollary 3.2.12. *If $f_1(x)$ is $O(g(x))$ and $f_2(x)$ is $O(g(x))$, then $(f_1 + f_2)(x)$ is $O(g(x))$.*

Theorem 3.2.13. *Suppose that $f_1(x)$ is $O(g_1(x))$ and that $f_2(x)$ is $O(g_2(x))$. Then $(f_1 f_2)(x)$ is $O(g_1(x)g_2(x))$.*

Proof. Suppose that $f_1(x)$ is $O(g_1(x))$ and that $f_2(x)$ is $O(g_2(x))$. Then $\exists C_1,\ \exists C_2,\ \exists k_1,\ \exists k_2$ such that

$$|f_1(x)| \leq C_1|g_1(x)| \quad \text{for } x > k_1$$

and

$$|f_2(x)| \leq C_2|g_2(x)| \quad \text{for } x > k_2.$$

Let $k = \max(k_1, k_2)$. Then

$$|f_1(x)f_2(x)| = |f_1(x)||f_2(x)| \leq C_1|g_1(x)|C_2|g_2(x)| \leq (C_1C_2)|g_1(x)g_2(x)| \qquad \text{for } x > k.$$

□

Remark 3.2.14. The goal in using big-O notation to estimate functions is to choose a function $g(x)$ as simple as possible, that grows relatively slowly so that $f(x)$ is $O(g(x))$.

Example 3.2.15. Give a big-O estimate for $f(n) = (2n + 1)\log(n!) + (n^2 + 3n - 1)\log n$, where n is a positive integer.

Proof. Since $(2n+1)$ is $O(n)$ and $\log(n!)$ is $O(n \log n)$, we have $(2n+1)\log(n!)$ is $O(n^2 \log n)$. Since n^2+3n-1 is $O(n^2)$, $(n^2+3n-1)\log n$ is $O(n^2 \log n)$. Therefore, $f(n) = (2n+1)\log(n!)+ (n^2 + 3n - 1)\log n$ is $O(n^2 \log n)$. □

Example 3.2.16. Give a big-O estimate for $f(x) = (x - 3)\log(x^2 + 4) + 5x^2$.

Proof. First of all,

$$\log(x^2 + 9) \leq \log(2x^2) = \log 2 + 2\log(x) \leq 3\log(x)$$

for $x > 3$. This shows that $\log(x^2+9)$ is $O(\log x)$. Thus $(x-3)\log(x^2+9)$ is $O(x \log x)$. Since $5x^2$ is $O(x^2)$, we have $f(x) = (x - 3)\log(x^2 + 4) + 5x^2$ is $O(\max(x \log x, x^2)) = O(x^2)$. □

Exercises

Page 216: 1-20, 25-27

Chapter 4

Number Theory and Cryptography

4.1 Divisibility and Modular Arithmetic

Division of an integer by a positive integer produces a quotient and a remainder. Working with these remainders leads to modular arithmetic, which plays an important role in mathematics and which is used throughout computer science. We will discuss some important applications of modular arithmetic later in this chapter.

4.1.1 Division

When one integer is divided by a second nonzero integer, the quotient may or may not be an integer.

Definition 4.1.1.

(1) If a and b are integers with $a \neq 0$, we say that a divides b if there is an integer c such that $b = ac$, or equivalently, if $\frac{b}{a}$ is an integer.

(2) When a divides b we say that a is a factor or divisor of b, and that b is a multiple of a.

(3) The notation $a \mid b$ denotes that a divides b. We write $a \nmid b$ when a does not divide b.

Remark 4.1.2. We can express $a \mid b$ using quantifiers as $(\exists c)(ac = b)$, where the universe of discourse is the set of integers.

Example 4.1.3. Let n and d be positive integers. How many positive integers not exceeding n are divisible by d?

Proof. The positive integers divisible by d are all the integers of the form dk, where k is a positive integer. Hence, the number of positive integers divisible by d that do not exceed n equals the number of integers k with $0 < dk \leq n$, or with $0 < k \leq n/d$. Therefore, there are $\lfloor n/d \rfloor$ positive integers not exceeding n that are divisible by d. □

Theorem 4.1.4. *(Basic properties of divisibility of integers)*
Let a, b, and c be integers, where $a \neq 0$. Then

(i) if $a \mid b$ and $a \mid c$, then $a \mid (b + c)$;

(ii) if $a \mid b$, then $a \mid bc$ for any integer c;

(iii) if $a \mid b$ and $b \mid c$, then $a \mid c$.

Proof. (i) Assume $a \mid b$ and $a \mid c$. Then there exist integers s and t such that $b = sa$ and $c = ta$. It follows that $b + c = (s + t)a$. Therefore, $a \mid (b + c)$.

(ii) Exercise.

(iii) Exercise.

□

Corollary 4.1.5. *Let a, b, and c be integers, where $a \neq 0$. If $a \mid b$ and $a \mid c$, then $a \mid (mb + nc)$ whenever m and n are integers.*

Proof. By part (ii) of Thm 1 we see that $a \mid mb$ and $a \mid nc$ whenever m and n are integers. By part (i) of Thm 1 it follows that $a \mid (mb + nc)$. □

4.1.2 The Division Algorithm

When an integer is divided by a positive integer, there is a quotient and a remainder, as the division algorithm shows.

Theorem 4.1.6. *(The Division Algorithm)*
Let a be an integer and d a positive integer. Then there are unique integers q and r, with $0 \leq r < d$, such that $a = dq + r$.

Definition 4.1.7. In the equality given in the division algorithm, d is called the divisor, a is called the dividend, q is called the quotient, and r is called the remainder. This notation is used to express the quotient and remainder:

$$q = a \text{ div } d, \qquad r = a \bmod d.$$

Remark 4.1.8. Note that both $q = a$ div d and $r = a \bmod d$ for a fixed d are functions on the set of integers. Furthermore, when a is an integer and d is a positive integer, we have a div $d = \lfloor a/d \rfloor$ and $a \bmod d = a - d\lfloor a/d \rfloor$.

Example 4.1.9. What are the quotient and remainder when 101 is divided by 9?

Proof. We have

$$101 = 11 \cdot 9 + 2.$$

Hence, the quotient when 101 is divided by 9 is $11 = 101$ div 9, and the remainder is $2 = 101 \bmod 9$.
OR $q = \lfloor a/d \rfloor = \lfloor \frac{101}{9} \rfloor = 11, \quad r = a - qd = 2.$ □

Example 4.1.10. What are the quotient and remainder when -45 is divided by 6?

Proof. We have

$$-45 = 6(-8) + 3.$$

Hence, the quotient when -45 is d[illegible]

Proof. (i) Assume $a \equiv b \pmod{m}$. Then $a - b = km$ for some $k \in \mathbb{Z}$. Assume $a \bmod m = r$. Then $a = qm + r$ with $0 \leq r < m$. Hence

$$b = a - km = qm + r - km = (q - k)m + r.$$

(ii) Assume $a \bmod m = b \bmod m$. Then there are q_1 and q_2 such that

$$a = q_1 m + r, \qquad b = q_2 m + r, \qquad 0 < r \leq m.$$

It follows that

$$a - b = (q_1 - q_2)m.$$

□

Remark 4.1.15. Recall that $a \bmod m$ and $b \bmod m$ are the remainders when a and b are divided by m, respectively. Consequently, Thm 3 also says that $a \equiv b \pmod{m}$ if and only if a and b have the same remainder when divided by m.

Example 4.1.16. Determine whether 27 is congruent to 3 modulo 6.

Proof. Because 6 divides $27 - 3 = 24$, we see that $27 \equiv 3 \pmod{6}$. □

Example 4.1.17. Determine whether 33 is congruent to 3 modulo 4.

Proof. Because 4 does not divide $33 - 3 = 30$, we see that $33 \not\equiv 3 \pmod{4}$. □

Example 4.1.18. List five integers that are congruent to 2 modulo 14.

Proof. $a \equiv 2 \pmod{14}$ if and only if $a \bmod 14 = 2 \bmod 14$. Thus $a \in \{2, 16, 30, 44, 58\}$. □

Theorem 4.1.19. *Let m be a positive integer. The integers a and b are congruent modulo m if and only if there is an integer k such that $a = b + km$.*

Proof. □

Theorem 4.1.20. *Let m be a positive integer. If $a \equiv b \ (mod\ m)$ and $c \equiv d \ (mod\ m)$, then*

$$a + c \equiv b + d \ (mod\ m)$$

and

$$ac \equiv bd \ (mod\ m).$$

Proof. □

Example 4.1.21. Because $11 \equiv 4 \ (\bmod\ 7)$ and $15 \equiv 1 \ (\bmod\ 7)$, it follows from Thm 5 that

$$26 \equiv 5 \ (\bmod\ 7)$$

and

$$165 \equiv 4 \ (\bmod\ 7).$$

Example 4.1.22. What time does a 12-hour clock read 70 hours after it reads 10 : 00?

Proof. $70 + 10 = 80 = 8 \bmod 12$ □

Remark 4.1.23. We must be careful working with congruences. Some properties we may expect to be true are not valid. For example, if $ac \equiv bc \pmod{m}$, the congruence $a \equiv b \pmod{m}$ may be false. Similarly, if $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, the congruence $a^c \equiv b^d \pmod{m}$ may be false.

The following corollary shows how to find the values of the mod m function at the sum and product of two integers using the values of this function at each of these integers.

Corollary 4.1.24. *Let m be a positive integer and let a and b be integers. Then*

$$(a + b) \bmod m = ((a \bmod m) + (b \bmod m)) \bmod m$$

and

$$(ab) \bmod m = ((a \bmod m)(b \bmod m)) \bmod m.$$

Proof. By the defns of (mod m) and of congruence modulo m, we know that $a \equiv (a \bmod m) (\bmod m)$ and $b \equiv (b \bmod m) (\bmod m)$. Hence, Thm 5 tells us that $a + b = ((a \bmod m) + (b \bmod m)) (\bmod m)$ and $ab = (a \bmod m)(b \bmod m) (\bmod m)$. The equalities in this corollary follow from these last two congruences by Thm 3. □

Example 4.1.25. Find $(179 \bmod 11 + 275 \bmod 11) \bmod 11$.

Proof. $(179 \bmod 11 + 275 \bmod 11) \bmod 11 = (179 + 275) \bmod 11 = 454 \bmod 11 = 3 \bmod 11$. □

Example 4.1.26. Find $(17 \bmod 7 \cdot 27 \bmod 7) \bmod 7$.

Proof. $(17 \bmod 7 \cdot 27 \bmod 7) \bmod 7 = (17 \cdot 27) \bmod 7 = 459 \bmod 7 = 4 \bmod 7$. □

4.1.4 Arithmetic Modulo m

We can define arithmetic operations on $\mathbb{Z}_m = \{0, 1, 2, \cdots, m - 1\}$, the set of nonnegative integers less than m.

Definition 4.1.27. Let $\mathbb{Z}_m = \{0, 1, 2, \cdots, m - 1\}$.

(1) We define addition modulo m on $\mathbb{Z}_m$, denoted by $+_m$ by

$$a +_m b = (a + b) \bmod m.$$

(2) We define multiplication modulo m on $\mathbb{Z}_m$, denoted by $\cdot_m$ by

$$a \cdot_m b = (a \cdot b) \bmod m.$$

Example 4.1.28. Use the defn of addition and multiplication in $\mathbb{Z}_m$ to find $7 +_5 9$ and $7 \cdot_5 9$.

Proof.

$$7 +_5 9 = (7 + 9) \bmod 5 = 16 \bmod 5 = 1,$$

and

$$7 \cdot_5 9 = (7 \cdot 9) \bmod 5 = 63 \bmod 5 = 3.$$

□

Example 4.1.29. Suppose that a and b are integers, $a \equiv 4 \pmod{13}$, and $b \equiv 9 \pmod{13}$. Find the integer c with $0 \leq c \leq 12$ such that $c \equiv a + b \pmod{13}$.

Proof. $c = a + b \pmod{13} = 4 + 9 \pmod{13} = 0 \pmod{13}$. □

Exercises

Page 244: 1-14, 20-33

4.2 Integer Representations and Algorithms

Integers can be expressed using any integer greater than one as a base, as we will show in this section. Although we commonly use decimal (base 10), representations, binary (base 2), octal (base 8), and hexadecimal (base 16) representations are often used, especially in computer science. Given a base b and an integer n, we will show how to construct the base b representation of this integer. We will also explain how to quickly covert between binary and octal and between binary and hexadecimal notations.

As mentioned in Section 3.1, the term algorithm originally referred to procedures for performing arithmetic operations using the decimal representations of integers. These algorithms, adapted for use with binary representations, are the basis for computer arithmetic. They provide good illustrations of the concept of an algorithm and the complexity of algorithms. For these reasons, they will be discussed in this section.

We will also introduce an algorithm for finding a div d and a mod d where a and d are integers with $d > 1$. Finally, we will describe an efficient algorithm for modular exponentiation, which is a particularly important algorithm for cryptography, as we will see in Section 4.6.

4.2.1 Representations of Integers

In everyday life we use decimal notation to express integers. For example, 965 is used to denote $9 \cdot 10^2 + 6 \cdot 10 + 5$. However, it is often convenient to use bases other than 10. In particular, computers usually use binary notation (with 2 as the base) when carrying out arithmetic, and octal (base 8) or hexadecimal (base 16) notation when expressing characters, such as letters or digits. In fact, we can use any integer greater than 1 as the base when expressing integers. This is stated in Thm 1.

Theorem 4.2.1. *Let b be an integer greater than 1. Then if n is a positive integer, it can be expressed uniquely in the form*

$$n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b + a_0,$$

where k is a nonnegative integer, $a_0, a_1, \cdots, a_k$ are nonnegative integers less than b, and $a_k \neq 0$.

Proof. A proof of this thm can be constructed using mathematical induction, a proof method that is discussed in Section 5.1. □

Remark 4.2.2. The representation of n given in Thm 1 is called the base b expansion of n. The base b expansion of n is denoted by $(a_k a_{k-1} \cdots a_1 a_0)_b$.

Remarks 4.2.3.

(1) Expansions with base $b = 2$ are called binary expansions. In binary notation each digit is either a 0 or a 1.

(2) Expansions with base $b = 8$ are called octal expansions. The octal digits are 0, 1, 2, 3, 4, 5, 6, 7.

(3) Expansions with base $b = 10$ are called decimal expansions.

(4) Expansions with base $b = 16$ are called hexadecimal expansions. Sixteen different digits are required for hexadecimal expansions. Usually, the hexadecimal digits used are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F, where the letters A through F represent the digits corresponding to the numbers 10 through 15 (in decimal notation).

Example 4.2.4. What is the decimal expansion of the integer 178.

Proof. $178 = 1 \cdot 10^2 + 7 \cdot 10 + 8$. Thus $178 = (178)_{10}$. □

Example 4.2.5. What is the decimal expansion of the integer that has $(1\ 0101\ 1111)_2$ as its binary expansion?

Proof. $(1\ 0101\ 1111)_2 = 1 \cdot 2^8 + 0 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2 + 1 = 256 + 64 + 16 + 8 + 4 + 2 + 1 = 351$. □

Example 4.2.6. What is the decimal expansion of the number with octal expansion $(7016)_8$?

Proof. $(7016)_8 = 7 \cdot 8^3 + 0 \cdot 8^2 + 1 \cdot 8 + 6 = 3589$. □

Example 4.2.7. What is the decimal expansion of the number with hexadecimal expansion $(2AE0B)_{16}$?

Proof. $(2AE0B)_{16} = 2 \cdot 16^4 + 10 \cdot 16^3 + 14 \cdot 16^2 + 0 \cdot 16 + 11 = 175627$. □

Base Conversion

We will now describe an algorithm for constructing the base b expansion of an integer n.

(1) First, divide n by b to obtain a quotient and remainder, that is,

$$n = bq_0 + a_0, \qquad 0 \leq a_0 < b.$$

The remainder, a_0, is the rightmost digit in the base b expansion of n.

(2) Next, divide q_0 by b to obtain

$$q_0 = bq_1 + a_1, \qquad 0 \leq a_1 < b.$$

We see that a_1 is the second digit from the right in the base b expansion of n.

(3) Continue this process, successively dividing the quotients by b, obtaining additional base b digits as the remainders.

(4) This process terminates when we obtain a quotient equal to zero.

It produces the base b digits of n from the right to the left.

Example 4.2.8. Find the octal expansion of $(12345)_{10}$.

Proof. First, divide 12345 by 8 to obtain

$$12345 = 8 \cdot 1543 + 1.$$

Successively dividing quotients by 8 gives

$$\begin{aligned} 1543 &= 8 \cdot 192 + 7, \\ 192 &= 8 \cdot 24 + 0, \\ 24 &= 8 \cdot 3 + 0, \\ 3 &= 8 \cdot 0 + 3. \end{aligned}$$

The successive remainders that we have found, 1, 7, 0, 0, and 3, are the digits from the right to the left of 12345 in base 8. Hence,

$$(12345)_{10} = (30071)_8.$$

□

Example 4.2.9. Find the hexadecimal expansion of $(177130)_{10}$.

Proof. Successively dividing quotients by 16 gives

$$\begin{aligned} 177130 &= 16 \cdot 11070 + 10, \\ 11070 &= 16 \cdot 691 + 14, \\ 691 &= 16 \cdot 43 + 3, \\ 45 &= 16 \cdot 2 + 11, \\ 2 &= 16 \cdot 0 + 2. \end{aligned}$$

Hence,

$$(177130)_{10} = (2B3EA)_{16}.$$

□

Example 4.2.10. Find the binary expansion of $(241)_{10}$.

Proof. Successively dividing quotients by 2 gives

$$\begin{aligned}
241 &= 2 \cdot 120 + 1,\\
120 &= 2 \cdot 60 + 0,\\
60 &= 2 \cdot 30 + 0,\\
30 &= 2 \cdot 15 + 0,\\
15 &= 2 \cdot 7 + 1\\
7 &= 2 \cdot 3 + 1\\
3 &= 2 \cdot 1 + 1\\
1 &= 2 \cdot 0 + 1.
\end{aligned}$$

Hence,

$$(241)_{10} = (1111\ 0001)_2.$$

□

4.2.2 Conversion Between Binary, Octal, And Hexadecimal Expansions

Conversion between binary and octal and between binary and hexadecimal expansions is extremely easy because each octal digit corresponds to a block of three binary digits and each hexadecimal digit corresponds to a block of four binary digits, with these correspondences shown in Table 1 without initial 0s shown.

TABLE 1: Hexadecimal, Octal, and Binary Representation of the Integers 0 through 15

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	1
Hexadecimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	
Octal	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	1
Binary	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	11

Example 4.2.11. (a) Find the octal and hexadecimal expansions of $(11111010111100)_2$.

(b) Find the binary expansions of $(765)_8$ and $(A8D)_{16}$.

Proof. (a) To convert $(11111010111100)_2$ into octal notation we group the binary digits into blocks of three, adding initial zeros at the start of the leftmost block if necessary. These blocks, from left to right, are 011, 111, 010, 111, and 100, corresponding to 3, 7, 2, 7, and 4, respectively. Consequently,

$$(11111010111100)_2 = (37274)_8.$$

To convert $(11111010111100)_2$ into hexadecimal notation we group the binary digits into blocks of four, adding initial zeros at the start of the leftmost block if necessary. These blocks, from left to right, are 0011, 1110, 1011, and 1100, corresponding to the hexadecimal digits 3, E, B, and C, respectively. Consequently,

$$(11111010111100)_2 = (3EBC)_{16}.$$

(b) To convert $(765)_8$ into binary notation, we replace each octal digit by a block of three binary digits. These blocks are 111, 110, and 101. Hence,

$$(765)_8 = (111110101)_2.$$

To convert $(A8D)_{16}$ into binary notation, we replace each hexadecimal digit by a block of four binary digits. These blocks are 1010, 1000, and 1101. Hence,

$$(A8D)_{16} = (101010001101)_2.$$

□

Exercises

Page 255: 1-12

4.3 Primes and Greatest Common Divisors

One important concept based on divisibility is that of a prime number. The study of prime numbers goes back to ancient times. Thousands of years ago it was known that there are infinitely many primes; the proof of this fact, found in the works of Euclid, is famous for its elegance and beauty.

We will discuss the distribution of primes among the integers. We will describe some of the results about primes found by mathematicians in the last 400 years. In particular, we will introduce an important thm, the fundamental thm of arithmetic. This thm, which asserts that every positive integer can be written uniquely as the product of primes in nondecreasing order, has many interesting consequences.

Primes have become essential in modern cryptographic systems, and we will develop some of their properties important in cryptography. For example, finding large primes is essential in modern cryptography. The length of time required to factor large integers into their prime factors is the basis for the strength of some important modern cryptographic systems.

In this section we will also study the greatest common divisor of two integers, as well as the least common multiple of two integers. We will develop an important algorithm for computing greatest common divisors, called the Euclidean algorithm.

4.3.1 Primes

Every integer greater than 1 is divisible by at least two integers, because a positive integer is divisible by 1 and by itself. Positive integers that have exactly two different positive integer factors are called primes.

Definition 4.3.1. (Primes)
An integer p greater than 1 is called prime if the only positive factors of p are 1 and p. A positive integer that is greater than 1 and is not prime is called composite.

Theorem 4.3.2 (The Fundamental Thm Of Arithmetic)**.** *Every integer greater than 1 can be written uniquely as a prime or as the product of two or more primes where the prime factors are written in order of nondecreasing size.*

Example 4.3.3. Find the prime factorization of 100

Proof. $100 = 2^2 \cdot 5^2$ □

4.3.2 Trial Division

It is often important to show that a given integer is prime. For instance, in cryptology, large primes are used in some methods for making messages secret. One procedure for showing that an integer is prime is based on the following observation.

Theorem 4.3.4. *If n is a composite integer, then n has a prime divisor less than or equal to $\sqrt{n}$.*

Proof. If n is composite, by the defn of a composite integer, we know that it has a factor a with $1 < a < n$. Hence, by the defn of a factor of a positive integer, we have $n = ab$, where b is a positive integer greater than 1. We will show that $a \leq \sqrt{n}$ or $b \leq \sqrt{n}$. If $a > \sqrt{n}$ and $b > \sqrt{n}$, then $ab > n$, which is a contradiction. Consequently, $a \leq \sqrt{n}$ or $b \leq \sqrt{n}$. Because both a and b are divisors of n, we see that n has a positive divisor not exceeding $\sqrt{n}$. This divisor is either prime or, by the fundamental thm of arithmetic, has a prime divisor less than itself. In either case, n has a prime divisor less than or equal to $\sqrt{n}$. □

From Thm 2, it follows that an integer is prime if it is not divisible by any prime less than or equal to its square root. This leads to the brute-force algorithm known as trial division. To use trial division we divide n by all primes not exceeding $\sqrt{n}$ and conclude that n is prime if it is not divisible by any of these primes.

Example 4.3.5. Show that 101 is prime.

Proof. The only primes not exceeding $\sqrt{101} < 11$ are 2, 3, 5, and 7. Because 101 is not divisible by 2, 3, 5, or 7 (the quotient of 101 and each of these integers is not an integer), it follows that 101 is prime. □

4.3.3 Finding prime factorization of an integer

Consider the problem of finding the prime factorization of n. Begin by dividing n by successive primes, starting with the smallest prime, 2. If n has a prime factor, then by Thm 3 a prime factor p not exceeding $\sqrt{n}$ will be found. So, if no prime factor not exceeding $\sqrt{n}$ is found, then n is prime. Otherwise, if a prime factor p is found, continue by factoring n/p. Note that n/p has no prime factors less than p. Again, if n/p has no prime factor greater than or equal to p and not exceeding its square root, then it is prime. Otherwise, if it has a prime factor q, continue by factoring $n/(pq)$. This procedure is continued until the factorization has been reduced to a prime.

Example 4.3.6. Find the prime factorization of 7007.

Proof. To find the prime factorization of 7007, first perform divisions of 7007 by successive primes, beginning with 2. None of the primes 2, 3, and 5 divides 7007. However, 7 divides 7007, with $7007/7 = 1001$.

Next, divide 1001 by successive primes, beginning with 7. It is immediately seen that 7 also divides 1001, because $1001/7 = 143$. Continue by dividing 143 by successive primes, beginning with 7. Although 7 does not divide 143, 11 does divide 143, and $143/11 = 13$. Because 13 is prime, the procedure is completed. It follows that

$$7007 = 7 \cdot 1001 = 7 \cdot 7 \cdot 143 = 7 \cdot 7 \cdot 11 \cdot 13.$$

Consequently, the prime factorization of 7007 is $7^2 \cdot 11 \cdot 13$. □

Example 4.3.7. Find the prime factorization of $909,090$.

Proof. First perform divisions of $909,090$ by successive primes, beginning with 2.

$$\begin{aligned}
\tfrac{909,090}{2} &= 454545,\\
\tfrac{454545}{3} &= 151515,\\
\tfrac{151515}{3} &= 50505,\\
\tfrac{50505}{3} &= 16835,\\
\tfrac{16835}{5} &= 3357,\\
\tfrac{3367}{7} &= 481,\\
\tfrac{481}{13} &= 37.
\end{aligned}$$

Thus

$$909,090 = 2 \cdot 3^3 \cdot 5 \cdot 7 \cdot 13 \cdot 37.$$

□

4.3.4 Greatest Common Divisors and Least Common Multiples

The largest integer that divides both of two integers is called the greatest common divisor of these integers and the smallest positive integer that is divisible by both of two integers is called the least common multiple of these integers.

Definition 4.3.8.

(1) Let a and b be integers, not both zero. The largest integer d such that $d|a$ and $d|b$ is called the greatest common divisor of a and b. The greatest common divisor of a and b is denoted by $\gcd(a, b)$.

(2) Let a and b be positive integers. The least common multiple of a and b is the smallest positive integer that is divisible by both a and b. The least common multiple of a and b is denoted by $\text{lcm}(a, b)$.

4.3.5 Finding the greatest common divisor and the least common multiple

One way to find the greatest common divisor and the least common multiple of two positive integers is to use the prime factorizations of these integers. Suppose that the prime factorizations of the positive integers a and b are

$$a = p_1^{a_1} p_2^{a_2} \cdots p_n^{a_n}, \qquad b = p_1^{b_1} p_2^{b_2} \cdots p_n^{b_n},$$

where each exponent is a nonnegative integer, and where all primes occurring in the prime factorization of either a or b are included in both factorizations, with zero exponents if necessary. Then

$$\gcd(a, b) = p_1^{\min(a_1, b_1)} p_2^{\min(a_2, b_2)} \cdots p_n^{\min(a_n, b_n)}$$

and

$$\operatorname{lcm}(a, b) = p_1^{\max(a_1, b_1)} p_2^{\max(a_2, b_2)} \cdots p_n^{\max(a_n, b_n)}.$$

Example 4.3.9. Find $\gcd(a, b)$ and $\operatorname{lcm}(a, b)$ if $a = 2^7 \cdot 5^3 \cdot 7^2,\ b = 3^{11} \cdot 5^5 \cdot 7$.

Proof. □

Example 4.3.10. Find $\gcd(120, 500)$ and $\operatorname{lcm}(120, 500)$.

Proof. We have $120 = 2^3 \cdot 3 \cdot 5, \qquad 500 = 2^2 \cdot 5^3$. Thus $\gcd(120, 500) = 2^2 \cdot 5 = 20$ and $\operatorname{lcm}(120, 300) = 2^3 \cdot 3 \cdot 5^3 = 3000$. □

Theorem 4.3.11 (Relationship between the greatest common divisor and least common multiple)**.** *Let a and b be positive integers. Then*

$$ab = \gcd(a, b) \cdot lcm(a, b).$$

Definition 4.3.12. (Relatively prime)

(a) The integers a and b are relatively prime if $\gcd(a, b) = 1$.

(b) The integers $a_1, a_2, \cdots, a_n$ are pairwise relatively prime if $\gcd(a_i, a_j) = 1$ for $1 = i < j \leq n$.

Example 4.3.13. Determine whether the integers 10, 17, and 21 are pairwise relatively prime.

Proof. Since $\gcd(10, 17) = 1,\ \gcd(10, 21) = 1$, and $\gcd(17, 21) = 1$, we conclude that 10, 17, and 21 are pairwise relatively prime. □

Example 4.3.14. Determine whether the integers 10, 19, and 24 are pairwise relatively prime.

Proof. Since $\gcd(10, 24) = 2 > 1$, we conclude that 10, 19, and 24 are not pairwise relatively prime. □

Exercises

Page 272: 1-30

4.4 Cryptography

Number theory plays a key role in cryptography, the subject of transforming information so that it cannot be easily recovered without special knowledge. Number theory is the basis of many classical ciphers, first used thousands of years ago, and used extensively until the 20th century.

4.4.1 Classical Cryptography

One of the earliest known uses of cryptography was by Julius Caesar. He made messages secret by shifting each letter three letters forward in the alphabet (sending the last three letters of the alphabet to the first three). For instance, using this scheme the letter B is sent to E and the letter X is sent to A. This is an example of encryption, that is, the process of making a message secret.

To express Caesar's encryption process mathematically, first replace each letter by an element of $\mathbb{Z}_{26}$, that is, an integer from 0 to 25 equal to one less than its position in the alphabet. Caesar's encryption method can be represented by the function f that assigns to the nonnegative integer $p \leq 25$, the integer $f(p)$ in the set $\{0, 1, 2, \ldots, 25\}$ with

$$f(p) = (p + 3) \bmod 26.$$

In the encrypted version of the message, the letter represented by p is replaced with the letter represented by $(p + 3) \bmod 26$.

0	1	2	3	4	5	6	7	8	9	10	11	12
A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

To recover the original message from a secret message encrypted by the Caesar cipher, the function f^{-1}, the inverse of f, is used. Note that the function f^{-1} sends an integer p from $\mathbb{Z}_{26}$, to

$$f^{-1}(p) = (p - 3) \bmod 26.$$

In other words, to find the original message, each letter is shifted back three letters in the alphabet, with the first three letters sent to the last three letters of the alphabet. The process of determining the original message from the encrypted message is called decryption.

Example 4.4.1. What is the secret message produced from the message "MEET YOU IN THE PARK" using the Caesar cipher?

Proof. First replace the letters in the message with numbers. This produces

$$12\ 4\ 4\ 19 \quad 24\ 14\ 20 \quad 8\ 13 \quad 19\ 7\ 4 \quad 15\ 0\ 17\ 10.$$

Now replace each of these numbers p by $f(p) = (p + 3) \bmod 26$. This gives

$$15\ 7\ 7\ 22 \quad 1\ 17\ 23 \quad 11\ 16 \quad 22\ 10\ 7 \quad 18\ 3\ 20\ 13.$$

Translating this back to letters produces the encrypted message "PHHW BRX LQ WKH SDUN." □

4.4.2 Generalization the Caesar cipher

There are various ways to generalize the Caesar cipher. For example, instead of shifting the numerical equivalent of each letter by 3, we can shift the numerical equivalent of each letter by k, so that

$$f(p) = (p + k) \bmod 26.$$

Such a cipher is called a shift cipher. Note that decryption can be carried out using

$$f^{-1}(p) = (p - k) \bmod 26.$$

Here the integer k is called a key.

Example 4.4.2. Encrypt the plain text message "STOP GLOBAL WARMING" using the shift cipher with shift $k = 11$.

Proof. First replace the letters in the message with numbers. This produces

$$18\ 19\ 14\ 15 \quad 6\ 11\ 14\ 1\ 0\ 11 \quad 22\ 0\ 17\ 12\ 8\ 13\ 6.$$

Now replace each of these numbers p by $f(p) = (p + 11) \bmod 26$. This gives

$$3\ 4\ 25\ 0 \quad 17\ 22\ 25\ 12\ 11\ 22 \quad 7\ 11\ 2\ 23\ 18\ 24\ 17.$$

Translating this last string back to letters, we obtain the cipher text "DEZA RWZMLW HLCXTYR." □

Example 4.4.3. Decrypt the cipher text message "LEWLYPLUJL PZ H NYLHA ALHJOLY" that was encrypted with the shift cipher with shift $k = 7$.

Proof. First replace the letters in the message with numbers. This produces

$$11\ 4\ 22\ 11\ 24\ 15\ 11\ 20\ 9\ 11 \quad 15\ 25 \quad 7 \quad 13\ 24\ 11\ 7\ 0 \quad 0\ 11\ 7\ 9\ 14\ 11\ 24.$$

Now replace each of these numbers p by $f^{-1}(p) = (p - 7) \bmod 26$. This gives

$$4\ 19\ 15\ 4\ 17\ 8\ 4\ 13\ 2\ 4 \quad 8\ 18 \quad 0 \quad 6\ 17\ 4\ 0\ 19 \quad 19\ 4\ 0\ 2\ 7\ 4\ 17.$$

Translating this last string back to letters, we obtain the cipher text "EXPERIENCE IS A GREAT TEACHER." □

4.4.3 The RSA Cryptosystem

In 1976, three researchers at the Massachusetts Institute of Technology-Ronald Rivest, Adi Shamir, and Leonard Adleman-introduced to the world a public key cryptosystem, known as the RSA system, from the initials of its inventors. As often happens with cryptographic discoveries, the RSA system had been discovered several years earlier in secret government research in the United Kingdom. Clifford Cocks, working in secrecy at the United Kingdom's Government Communications Headquarters (GCHQ), had discovered this cryptosystem in 1973. However, his invention was unknown to the outside world until the late 1990s, when he was allowed to share classified GCHQ documents from the early 1970s.

In the RSA cryptosystem, each individual has an encryption key (n, e) where $n = pq$, the modulus is the product of two large primes p and q, say with 200 digits each, and an exponent e that is relatively prime to $(p - 1)(q - 1)$. To produce a usable key, two large primes must be found. This can be done quickly on a computer using probabilistic primality tests, referred to earlier in this section. However, the product of these primes $n = pq$, with approximately 400 digits, cannot, as far as is currently known, be factored in a reasonable length of time. As we will see, this is an important reason why decryption cannot, as far as is currently known, be done quickly without a separate decryption key.

4.4.4 RSA Encryption

To encrypt messages using a particular key (n, e), we first translate a plaintext message M into sequences of integers. To do this, we first translate each plaintext letter into a two-digit number, using the same translation we employed for shift ciphers, with one key difference. That is, we include an initial zero for the letters A through J, so that A is translated into 00, B into $01, \cdots$, and J into 09. Then, we concatenate these two-digit numbers into strings of digits. Next, we divide this string into equally sized blocks of $2N$ digits, where $2N$ is the largest even number such that the number $2525 \cdots 25$ with $2N$ digits does not exceed n. (When necessary, we pad the plaintext message with dummy Xs to make the last block the same size as all other blocks.)

After these steps, we have translated the plaintext message M into a sequence of integers $m_1, m_2, \cdots, m_k$ for some integer k. Encryption proceeds by transforming each block m_i to a ciphertext block c_i. This is done using the function

$$C = M^e \bmod n.$$

(To perform the encryption, we use an algorithm for fast modular exponentiation, such as Algorithm 5 in Section 4.2.) We leave the encrypted message as blocks of numbers and send these to the intended recipient. Because the RSA cryptosystem encrypts blocks of characters into blocks of characters, it is a block cipher.

Exercises

Page 304: 1-32

Chapter 5

Induction and Recursion

5.1 Mathematical Induction

Proof by mathematical induction is a very useful method in proving the validity of a mathematical statement $(\forall n)P(n)$ involving integers n greater than or equal to some initial integer n_0.

Suppose that we have an infinite ladder, and we want to know whether we can reach every step on this ladder. We know two things:

(1) We can reach the first rung of the ladder.

(2) If we can reach a particular rung of the ladder, then we can reach the next rung.

Can we conclude that we can reach every rung? By (1), we know that we can reach the first rung of the ladder. Moreover, because we can reach the first rung, by (2), we can also reach the second rung; it is the next rung after the first rung. Applying (2) again, because we can reach the second rung, we can also reach the third rung. Continuing in this way, we can show that we can reach the fourth rung, the fifth rung, and so on.

Another way to illustrate the principle of mathematical induction is to consider an infinite row of dominoes, labeled $1, 2, 3, \cdots, n, \cdots$, where each domino is standing up. Let $P(n)$ be the proposition that domino n is knocked over. If the first domino is knocked over-i.e., if $P(1)$ is true-and if, whenever the kth domino is knocked over, it also knocks the $(k+1)$st domino over, then all the dominoes are knocked over.

5.1.1 Principle of mathematical induction

If $P(n)$ is a propositional function involving the positive integer n such that

(1) $P(1)$ is true, and

(2) $P(k)$ is true $\rightarrow P(k+1)$ is true for any arbitrary natural number k,

then $P(n)$ is true for every natural number n; that is "$(\forall n)P(n)$ is true".

Remarks 5.1.1.

(1) The condition (1) is called the base step and its verification, that is usually easy, assure us that the theorem is true for at least the case $n = 1$.

(2) The condition (2) is called the inductive step. To verify this condition, we must prove an auxiliary theorem whose hypotheses is "$P(k)$ is true" and whose conclusion is "$P(k+1)$ is true". The hypotheses is called the inductive hypotheses.

Example 5.1.2. Show that if n is a positive integer, then

$$1 + 2 + \cdots + n = \frac{n(n+1)}{2}.$$

Proof.

$$P(n) : \sum_{j=1}^{n} j = 1 + 2 + \cdots + n = \frac{n(n+1)}{2}.$$

(1) BASIS STEP: $P(1) :$ $\sum_{j=1}^{1} j = 1 = \frac{1(2)}{2}$. Thus $P(1)$ is true.

(2) INDUCTIVE STEP: Assume that $P(k)$ is true; that is

$$\sum_{j=1}^{k} j = \frac{k(k+1)}{2}.$$

We want to show that $P(k+1)$ is true, namely, that

$$\sum_{j=1}^{k+1} j = \frac{(k+1)(k+2)}{2}.$$

Now

$$\sum_{j=1}^{k+1} j = \sum_{j=1}^{k} j + (k+1) = \frac{k(k+1)}{2} + (k+1) = \frac{(k+1)(k+2)}{2}.$$

Therefore, by mathematical induction we know that $P(n)$ is true for all positive integers n.

□

Example 5.1.3. Use mathematical induction to prove that

$$\forall n \in \mathbb{Z}^+, \quad \sum_{j=1}^{n} (3j - 2) = \frac{1}{2} n(3n - 1).$$

Proof.

$$P(n) : \sum_{j=1}^{n} (3j - 2) = \frac{1}{2} n(3n - 1).$$

$$(1)\ P(1): \qquad \sum_{j=1}^{1}(3j-2) = \frac{1}{2}(1)(3-1).$$

Thus $P(1)$ is true. (2) Assume that $P(k)$ is true; that is

$$\sum_{j=1}^{k}(3j-2) = \frac{1}{2}k(3k-1).$$

Then

$$\begin{aligned}\sum_{j=1}^{k+1}(3j-2) &= \sum_{j=1}^{k}(3j-2) + (3k+3-2)\\ &= \frac{1}{2}k(3k-1) + 3k + 1 = \frac{1}{2}[3k^2+5k+2]\\ &= \frac{1}{2}(k+1)(3k+2).\end{aligned}$$

□

Example 5.1.4. Prove that $\forall n \in \mathbb{Z}^+$,

$$3 + 11 + 19 + \cdots + 8n - 5 = 4n^2 - n.$$

Proof. □

Example 5.1.5. Prove that for all positive integers n, $3^n \geq 2^n + 1$.

Proof. □

Example 5.1.6. Show that $n! \geq 2^n, \quad \forall n \geq 4$.

Proof. □

Exercises

Page 329: 3-30

Chapter 6

Counting

6.1 The Basics of Counting

Suppose that a password on a computer system consists of six, seven, or eight characters. Each of these characters must be a digit or a letter of the alphabet. Each password must contain at least one digit. How many such passwords are there? The techniques needed to answer this question and a wide variety of other counting problems will be introduced in this section.

Counting problems arise throughout mathematics and computer science. For example, we must count the successful outcomes of experiments and all the possible outcomes of these experiments to determine probabilities of discrete events. We need to count the number of operations used by an algorithm to study its time complexity.

We will introduce the basic techniques of counting in this section. These methods serve as the foundation for almost all counting techniques.

Basic Counting Principles

We first present two basic counting principles, the product rule and the sum rule. Then we will show how they can be used to solve many different counting problems.

The product rule applies when a procedure is made up of separate tasks.

The Product Rule

Suppose that a procedure can be broken down into a sequence of two tasks. If there are n_1 ways to do the first task and for each of these ways of doing the first task, there are n_2 ways to do the second task, then there are $n_1 n_2$ ways to do the procedure.

Example 6.1.1. A new company with just two employees, Sanchez and Patel, rents a floor of a building with 12 offices. How many ways are there to assign different offices to these two employees?

Proof. The procedure of assigning offices to these two employees consists of assigning an office to Sanchez, which can be done in 12 ways, then assigning an office to Patel different

from the office assigned to Sanchez, which can be done in 11 ways. By the product rule, there are

$$12 \cdot 11 = 132$$

ways to assign offices to these two employees. □

Example 6.1.2. The chairs of an auditorium are to be labeled with an uppercase English letter followed by a positive integer not exceeding 100. What is the largest number of chairs that can be labeled differently?

Proof. The procedure of labeling a chair consists of two tasks, namely, assigning to the seat one of the 26 uppercase English letters, and then assigning to it one of the 100 possible integers. The product rule shows that there are

$$26 \cdot 100 = 2600$$

different ways that a chair can be labeled. Therefore, the largest number of chairs that can be labeled differently is 2600. □

Example 6.1.3. There are 32 microcomputers in a computer center. Each microcomputer has 24 ports. How many different ports to a microcomputer in the center are there?

Proof. The procedure of choosing a port consists of two tasks, first picking a microcomputer and then picking a port on this microcomputer. Because there are 32 ways to choose the microcomputer and 24 ways to choose the port no matter which microcomputer has been selected, the product rule shows that there are

$$32 \cdot 24 = 768$$

ports. □

Generalized Product Rule

An extended version of the product rule is often useful. Suppose that a procedure is carried out by performing the tasks $T_1, T_2, \ldots, T_m$ in sequence. If each task $T_i, i = 1, 2, \ldots, m$, can be done in n_i ways, regardless of how the previous tasks were done, then there are $n_1 \cdot n_2 \cdots n_m$ ways to carry out the procedure. This version of the product rule can be proved by mathematical induction from the product rule for two tasks.

Example 6.1.4. How many different bit strings of length seven are there?

Proof. Each of the seven bits can be chosen in two ways, because each bit is either 0 or 1. Therefore, the product rule shows there are a total of $2^7 = 128$ different bit strings of length seven. □

Example 6.1.5. How many different license plates can be made if each plate contains a sequence of three uppercase English letters followed by three digits (and no sequences of letters are prohibited, even if they are obscene)?

Proof. There are 26 choices for each of the three uppercase English letters and ten choices for each of the three digits. Hence, by the product rule there are a total of

$$26 \cdot 26 \cdot 26 \cdot 10 \cdot 10 \cdot 10 = 17,576,000$$

possible license plates. □

Example 6.1.6. How many functions are there from a set with m elements to a set with n elements?

Proof. A function corresponds to a choice of one of the n elements in the codomain for each of them elements in the domain. Hence, by the product rule there are

$$n \cdot n \cdots n = n^m$$

functions from a set with m elements to one with n elements. For example, there are $5^3 = 125$ different functions from a set with three elements to a set with five elements. □

Example 6.1.7. How many one-to-one functions are there from a set with m elements to one with n elements?

Proof. First note that when $m > n$ there are no one-to-one functions from a set with m elements to one with n elements.

Now let $m \leq n$. Suppose the elements in the domain are $a_1, a_2, \cdots, a_m$. There are n ways to choose the value of the function at a_1. Because the function is one-to-one, the value of the function at a_2 can be picked in $n - 1$ ways (because the value used for a_1 cannot be used again). In general, the value of the function at a_k can be chosen in $n - k + 1$ ways. By the product rule, there are

$$n(n-1)(n-2)\cdots(n-m+1)$$

one-to-one functions from a set with m elements to one with n elements.

For example, there are

$$5 \cdot 4 \cdot 3 = 60$$

one-to-one functions from a set with three elements to a set with five elements. □

The Sum Rule

If a task can be done either in one of n_1 ways or in one of n_2 ways, where none of the set of n_1 ways is the same as any of the set of n_2 ways, then there are $n_1 + n_2$ ways to do the task.

Example 6.1.8. Suppose that either a member of the mathematics faculty or a student who is a mathematics major is chosen as a representative to a university committee. How many different choices are there for this representative if there are 37 members of the mathematics faculty and 83 mathematics majors and no one is both a faculty member and a student?

Proof. There are 37 ways to choose a member of the mathematics faculty and there are 83 ways to choose a student who is a mathematics major. Choosing a member of the mathematics faculty is never the same as choosing a student who is a mathematics major because no one is both a faculty member and a student. By the sum rule it follows that there are

$$37 + 83 = 120$$

possible ways to pick this representative. □

We can extend the sum rule to more than two tasks. Suppose that a task can be done in one of n_1 ways, in one of n_2 ways, ..., or in one of nm ways, where none of the set of n_i ways of doing the task is the same as any of the set of n_j ways, for all pairs i and j with $1 \leq i < j \leq m$. Then the number of ways to do the task is

$$n_1 + n_2 + \cdots + n_m.$$

This extended version of the sum rule is often useful in counting problems, as Examples 13 and 14 show. This version of the sum rule can be proved using mathematical induction from the sum rule for two sets.

Example 6.1.9. A student can choose a computer project from one of three lists. The three lists contain 23, 15, and 19 possible projects, respectively. No project is on more than one list. How many possible projects are there to choose from?

Proof. The student can choose a project by selecting a project from the first list, the second list, or the third list. Because no project is on more than one list, by the sum rule there are

$$23 + 15 + 19 = 57$$

ways to choose a project. □

Example 6.1.10.

Proof. □

Exercises

Page 396: 1-16

Chapter 7

Chapter 8

Chapter 9

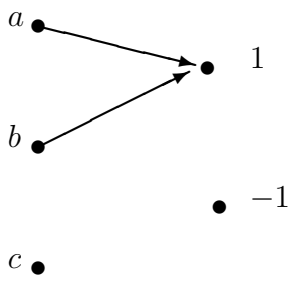
Relations

9.1 Relations and Their Properties

The most direct way to express a relationship between elements of two sets is to use ordered pairs made up of two related elements. For this reason, sets of ordered pairs are called binary relations. In this section we introduce the basic terminology used to describe binary relations. Later in this chapter we will use relations to solve problems involving communications networks, project scheduling, and identifying elements in sets with common properties.

Definition 9.1.1. (Binary relations)
A binary relation R from a set A to a set B is a subset of the Cartesian product $A \times B$.

Notation 9.1.2. Let R be a relation from a set



Another way to represent this relation is to use a table.

R	1	-1
a	$\times$	
b	$\times$	
c		

We will discuss representations of relations in more detail in Section 9.3.

Remark 9.1.5. If R is a relation from A to A, then R is called a relation on A.

Example 9.1.6. Let $A = \{1, 2, 3\}$ and let $R = \{(1, 1), (2, 2), (1, 3)\}$. Then R is a relation from A to A.

Example 9.1.7. Let A be the set $\{1, 2, 3, 4\}$. Which ordered pairs are in the relation $R = \{(a, b)|\ a \text{ divides } b\}$?

Proof. Because $(a, b) \in R$ if and only if $a, b \in A$ such that a divides b, we see that

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

□

Example 9.1.8. Consider these relations on the set of integers:

$$\begin{aligned}
R_1 &= \{(a, b)|\ a \leq b\},\\
R_2 &= \{(a, b)|\ a > b\},\\
R_3 &= \{(a, b)|\ a = b \text{ or } a = -b\},\\
R_4 &= \{(a, b)|\ a = b\},\\
R_5 &= \{(a, b)|\ a = b + 1\},\\
R_6 &= \{(a, b)|\ a + b \leq 3\}.
\end{aligned}$$

Which of these relations contain each of the pairs $(1, 1), (1, 2), (2, 1), (1, -1)$, and $(2, 2)$?

Proof. The pair $(1, 1)$ is in R_1, R_3, R_4, and R_6; $(1, 2)$ is in R_1 and R_6; $(2, 1)$ is in R_2, R_5, and R_6; $(1, -1)$ is in R_2, R_3, and R_6; and finally, $(2, 2)$ is in R_1, R_3, and R_4. □

9.1.1 Properties of Relations

There are several properties that are used to classify relations on a set. We will introduce the most important of these here.

9.1.2 Reflexive Relations

In some relations an element is always related to itself. For instance, let R be the relation on the set of all people consisting of pairs (x, y) where x and y have the same mother and the same father. Then $x\ R\ x$ for every person x.

Definition 9.1.9. (Reflexive relations)
A relation R on a set A is called reflexive if $x\ R\ x$ for all $x \in A$.

Example 9.1.10. Let $A = \{1, 2\}$ and let R be a relation on A given by $R = \{(1, 1), (1, 2), (2, 2)\}$. Show that R is reflexive.

Proof. $(1, 1) \in R$ and $(2, 2) \in R$ implies R is reflexive. □

Example 9.1.11. Let $A = \{a, b, c\}$ and let R be a relation on A given by $R = \{(a, b), (b, a), (a, c), (c, a), (b, b$ Show that R is not reflexive.

Proof. $c \in A$ but $(c, c) \notin R$. Thus R is NOT reflexive. □

Example 9.1.12. Let $R = \{(x, y) \in \mathbb{Z} \times \mathbb{Z} : \quad xy > 0\}$. Determine whether R is reflexive or not.

Proof. $0 \in \mathbb{Z}$ but $(0, 0) \notin R \Rightarrow R$ is not reflexive. □

Example 9.1.13. Let R be the relation on the set of all integers $\mathbb{Z}$ defined by $x, y) \in R$ if and only if $x \equiv y(mod 3)$. Determine whether R is reflexive or not.

Proof. □

9.1.3 Symmetric and Antisymmetric Relations

In some relations an element is related to a second element if and only if the second element is also related to the first element. The relation consisting of pairs (x, y), where x and y are students at your school with at least one common class has this property. Other relations have the property that if an element is related to a second element, then this second element is not related to the first. The relation consisting of the pairs (x, y), where x and y are students at your school, where x has a higher grade point average than y has this property.

Definition 9.1.14. (Symmetric relations)
Let R be a relation on a set A.

(1) R is called symmetric if for all $x, y \in A$, $x\ R\ y \rightarrow y\ R\ x$.

(2) R is called antisymmetric if for all $x, y \in A$, $(x\ R\ y \wedge y\ R\ x) \rightarrow x = y$.

Example 9.1.15. Let $A = \{1, 2\}$ and let R be a relation on A given by $R = \{(1, 1), (1, 2), (2, 2)\}$. Show that R is not symmetric.

Proof. $(1, 2) \in R$ but $(2, 1) \notin R \Rightarrow R$ is NOT symmetric. □

Example 9.1.16. Let $A = \{a, b, c\}$ and let R be a relation on A given by $R = \{(a, b), (b, a), (a, c), (c, a), (b, b$
Show that R is symmetric.

Proof. $(a, b) \in R \wedge (b, a) \in R$
$(a, c) \in R \wedge (c, a) \in R$.
Thus R is symmetric. □

Example 9.1.17. Which of the following relations are symmetric and which are antisymmetric?

$$R_1 = \{(a, b) |\ a \leq b\},$$
$$R_2 = \{(a, b) |\ a > b\},$$
$$R_3 = \{(a, b) |\ a = b \text{ or } a = -b\},$$
$$R_4 = \{(a, b) |\ a = b\},$$
$$R_5 = \{(a, b) |\ a = b + 1\},$$
$$R_6 = \{(a, b) |\ a + b \leq 3\}.$$

Proof. The relations R_3, R_4, and R_6 are symmetric. R_3 is symmetric, for if $a = b$ or $a = -b$, then $b = a$ or $b = -a$. R_4 is symmetric because $a = b$ implies that $b = a$. R_6 is symmetric because $a + b \leq 3$ implies that $b + a \leq 3$. None of the other relations is symmetric.

The relations R_1, R_2, R_4, and R_5 are antisymmetric. R_1 is antisymmetric because the inequalities $a \leq b$ and $b \leq a$ imply that $a = b$. R_2 is antisymmetric because it is impossible that $a > b$ and $b > a$. R_4 is antisymmetric, because two elements are related with respect to R_4 if and only if they are equal. R_5 is antisymmetric because it is impossible that $a = b + 1$ and $b = a+1$. The reader should verify that none of the other relations is antisymmetric. □

Example 9.1.18. Is the "divides" relation on the set of positive integers symmetric? Is it antisymmetric?

Proof. This relation is not symmetric because $1 \mid 2$, but $2 \not\mid 1$.
It is antisymmetric, for if a and b are positive integers with $a \mid b$ and $b \mid a$, then $a = b$. □

9.1.4 Transitive Relations

Let R be the relation consisting of all pairs (x, y) of students at your school, where x has taken more credits than y. Suppose that x is related to y and y is related to z. This means that x has taken more credits than y and y has taken more credits than z. We can conclude that x has taken more credits than

Example 9.1.21. Let $A = \{a, b, c\}$ and let R be a relation on A given by $R = \{(a, b), (b, a), (a, c), (c, a), (b, b$ Show that R is not transitive.

Proof. $(a, b) \in R \wedge (b, a) \in R$ but $(a, a) \notin R$. Thus R is NOT transitive. □

Example 9.1.22. Let $R = \{(x, y) \in \mathbb{Z} \times \mathbb{Z} : \quad xy > 0\}$. Determine whether R is transitive or not.

Proof. Let $(x, y) \in R \wedge (y, z) \in R \Rightarrow xy > 0 \wedge yz > 0 \Rightarrow xzy^2 > 0 \Rightarrow xz > 0$.
Therefore $(x, z) \in R$ and it follows that R is transitive.
Alternative solution:
If $y < 0$, then $x < 0$ and $z < 0$. Hence $xz > 0$.
If $y > 0$, then $x > 0$ and $z > 0$. Hence $xz > 0$.

□

Example 9.1.23. Let R be the relation on the set of all integers $\mathbb{Z}$ defined by $x, y) \in R$ if and only if $x \equiv y(mod 4)$. Determine whether R is transitive or not.

Proof. □

9.1.5 Combining Relations

Because relations from A to B are subsets of $A \times B$, two relations from A to B can be combined in any way two sets can be combined.

Example 9.1.24. Let $A = \{1, 2, 3\}$ and $B = \{1, 2, 3, 4\}$. The relations $R_1 = \{(1, 1), (2, 2), (3, 3)\}$ and $R_2 = \{(1, 1), (1, 2), (1, 3), (1, 4)\}$ can be combined to obtain

$$R_1 \cup R_2 = \{(1, 1), (1, 2), (1, 3), (1, 4), (2, 2), (3, 3)\},$$
$$R_1 \cap R_2 = \{(1, 1)\},$$
$$R_1 - R_2 = \{(2, 2), (3, 3)\},$$
$$R_2 - R_1 = \{(1, 2), (1, 3), (1, 4)\}.$$

Example 9.1.25. Let R_1 be the "less than" relation on the set of real numbers and let R_2 be the "greater than" relation on the set of real numbers, that is, $R_1 = \{(x, y)|x < y\}$ and $R_2 = \{(x, y)|x > y\}$. What are $R_1 \cup R_2, R_1 \cap R_2, R_1 - R_2, R_2 - R_1$?

Proof.

$$R_1 \cup R_2 = \{(x, y)|(x, y) \in R_1 \vee (x, y) \in R_2\} = \{(x, y)|x < y \vee y < x\} = \{(x, y)|x \neq y\}.$$

$$R_1 \cap R_2 = \{(x, y)|(x, y) \in R_1 \wedge (x, y) \in R_2\} = \{(x, y)|x < y \wedge y < x\} = \varnothing.$$

$$R_1 - R_2 = \{(x, y)|(x, y) \in R_1 \wedge (x, y) \notin R_2\} = \{(x, y)|x < y \wedge \neg(y < x)\} = \{(x, y)|x < y \wedge y \geq x\} = R_1.$$

$$R_2 - R_1 = \{(x, y)|(x, y) \in R_2 \wedge (x, y) \notin R_1\} = \{(x, y)|y < x \wedge \neg(x < y)\} = \{(x, y)|y < x \wedge x \geq y\} = R_2.$$

□

9.1.6 Composition of Relations

Definition 9.1.26. (Relation composition)
Let R be a relation from a set A to a set B and S a relation from B to a set C. The composite of R and S is the relation consisting of ordered pairs (a, c), where $a \in A,\ c \in C$, and for which there exists an element $b \in B$ such that $(a, b) \in R$ and $(b, c) \in S$. We denote the composite of R and S by $S \circ R$.

Remark 9.1.27. $S \circ R = \{(a, c) \in A \times C :\ (\exists b \in B)((a, b) \in R \wedge (b, c) \in S)\}$.

Computing the composite $S \circ R$ of two relations requires that we find elements that are the second element of ordered pairs in R and the first element of ordered pairs in S.

Example 9.1.28. What is the composite of the relations R and S, where R is the relation from $\{1, 2, 3\}$ to $\{1, 2, 3, 4\}$ with $R = \{(1, 1), (1, 4), (2, 3), (3, 1), (3, 4)\}$ and S is the relation from $\{1, 2, 3, 4\}$ to $\{0, 1, 2\}$ with $S = \{(1, 0), (2, 0), (3, 1), (3, 2), (4, 1)\}$?

Proof. $S \circ R$ is constructed using all ordered pairs in R and ordered pairs in S, where the second element of the ordered pair in R agrees with the first element of the ordered pair in S. Thus $S \circ R = \{(1, 0), (1, 1), (2, 1), (2, 2), (3, 0), (3, 1)\}$. □

Definition 9.1.29. (Relation powers)
Let R be a relation on a set A. The powers $R^n,\ n = 1, 2, 3, \cdots$, are defined recursively by

$$R^1 = R, \quad R^{n+1} = R^n \circ R.$$

Example 9.1.30. Let $R = \{(1, 1), (2, 1), (3, 2), (4, 3)\}$. Find the powers $R^n,\ n = 2, 3, 4, \cdots$.

Proof.

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

It also follows that $R^n = R^3$ for $n = 5, 6, 7, \cdots$. □

Exercises

Page 581: 1-10, 34-37, 40-43.

9.2 Omit

9.3 Representing Relations

There are many ways to represent a relation between finite sets. As we have seen in Section 9.1, one way is to list its ordered pairs. Another way to represent a relation is to use a table, as we did in Section 9.1. In this section we will discuss two alternative methods for representing relations. One method uses zero-one matrices. The other method uses pictorial

representations called directed graphs, which we will discuss later in this section. Generally, matrices are appropriate for the representation of relations in computer programs. On the other hand, people often find the representation of relations using directed graphs useful for understanding the properties of these relations.

9.3.1 Representing Relations Using Matrices

A relation between finite sets can be represented using a zero-one matrix. Suppose that R is a relation from $A = \{a_1, a_2, \cdots, a_m\}$ to $B = \{b_1, b_2, \cdots, b_n\}$. (Here the elements of the sets A and B have been listed in a particular, but arbitrary, order. Furthermore, when $A = B$ we use the same ordering for A and B.) The relation R can be represented by the $m \times n$ matrix $M_R = [m_{ij}]$, where

$$m_{ij} = \begin{cases} 1, & \text{if } (a_i, b_j) \in R \\ 0, & \text{if } (a_i, b_j) \notin R. \end{cases}$$

In other words, the zero-one matrix representing R has a 1 as its (i, j) entry when a_i is related to b_j, and a 0 in this position if a_i is not related to b_j. (Such a representation depends on the orderings used for A and B.)

Example 9.3.1. Suppose that $A = \{1, 2, 3\}$ and $B = \{1, 2\}$. Let R be the relation from A to B containing (a, b) if $a \in A$, $b \in B$, and $a > b$. What is the matrix representing R if $a_1 = 1$, $a_2 = 2$, and $a_3 = 3$, and $b_1 = 1$ and $b_2 = 2$?

Proof. Because $R = \{(2, 1), (3, 1), (3, 2)\}$, the matrix for R is

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

□

Example 9.3.2. Let $A = \{a_1, a_2, a_3\}$ and $B = \{b_1, b_2, b_3, b_4, b_5\}$. Which ordered pairs are in the relation R represented by the matrix

$$M_R = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}?$$

Proof. Because R consists of those ordered pairs (a_i, b_j) with $m_{ij} = 1$, it follows that $R = \{(a_1, b_2), (a_2, b_1), (a_2, b_3), (a_2, b_4), (a_3, b_1), (a_3, b_3), (a_3, b_5)\}$. □

9.3.2 Representing Relations and Properties of Relations

The matrix of a relation on a set, which is a square matrix, can be used to determine whether the relation has certain properties. Recall that a relation R on A is reflexive if $(a, a) \in R$ whenever $a \in A$. Thus,R is reflexive if and only if $(a_i, a_i) \in R$ for $i = 1, 2, \cdots, n$. Hence, R

is reflexive if and only if $m_{ii} = 1$, for $i = 1, 2, \cdots, n$. In other words, R is reflexive if all the elements on the main diagonal of M_R are equal to 1. Note that the elements off the main diagonal can be either 0 or 1.

The relation R is symmetric if $(a, b) \in R$ implies that $(b, a) \in R$. Consequently, the relation R on the set $A = \{a_1, a_2, \cdots, a_n\}$ is symmetric if and only if $(a_j, a_i) \in R$ whenever $(a_i, a_j) \in R$. In terms of the entries of M_R, R is symmetric if and only if $m_{ji} = 1$ whenever $m_{ij} = 1$. This also means $m_{ji} = 0$ whenever $m_{ij} = 0$. Consequently, R is symmetric if and only if $m_{ij} = m_{ji}$, for all pairs of integers i and j with $i = 1, 2, \cdots, n$ and $j = 1, 2, \cdots, n$. Recalling the defn of the transpose of a matrix from Section 2.6, we see that R is symmetric if and only if

$$M_R = (M_R)^t,$$

that is, if M_R is a symmetric matrix.

The relation R is antisymmetric if and only if $(a, b) \in R$ and $(b, a) \in R$ imply that $a = b$. Consequently, the matrix of an antisymmetric relation has the property that if $m_{ij} = 1$ with $i \neq j$, then $m_{ji} = 0$. Or, in other words, either $m_{ij} = 0$ or $m_{ji} = 0$ when $i \neq j$.

Example 9.3.3. Suppose that the relation R on a set is represented by the matrix

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

Is R reflexive, symmetric, and/or antisymmetric?

Proof. Because all the diagonal elements of this matrix are equal to 1, R is reflexive. Moreover, because M_R is symmetric, it follows that R is symmetric. It is also easy to see that R is not antisymmetric. □

9.3.3 Representing Combination of Relations

The Boolean operations join and meet (discussed in Section 2.6) can be used to find the matrices representing the union and the intersection of two relations. Suppose that R_1 and R_2 are relations on a set A represented by the matrices M_{R_1} and M_{R_2}, respectively. The matrix representing the union of these relations has a 1 in the positions where either M_{R_1} or M_{R_2} has a 1. The matrix representing the intersection of these relations has a 1 in the positions where both M_{R_1} and M_{R_2} have a 1. Thus, the matrices representing the union and intersection of these relations are

$$M_{R_1 \cup R_2} = M_{R_1} \vee M_{R_2} \quad \text{and} \quad M_{R_1 \cap R_2} = M_{R_1} \wedge M_{R_2}.$$

Example 9.3.4. Suppose that the relations R_1 and R_2 on a set A are represented by the matrices

$$M_{R_1} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{and} \quad M_{R_2} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}.$$

What are the matrices representing $R_1 \cup R_2$ and $R_1 \cap R_2$?

Proof. The matrices of these relations are

$$M_{R_1 \cup R_2} = M_{R_1} \vee M_{R_2} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix},$$

$$M_{R_1 \cap R_2} = M_{R_1} \wedge M_{R_2} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

□

9.3.4 Representing Composition of Relations

Suppose that A, B, and C have m, n, and p elements, respectively. Let the zero- one matrices for $S \circ R$, R, and S be $M_{S \circ R} = [t_{ij}]$, $M_R = [r_{ij}]$, and $M_S = [s_{ij}]$, respectively (these matrices have sizes $m \times p$, $m \times n$, and $n \times p$, respectively). The ordered pair (a_i, c_j) belongs to $S \circ R$ if and only if there is an element b_k such that (a_i, b_k) belongs to R and (b_k, c_j) belongs to S. It follows that $t_{ij} = 1$ if and only if $r_{ik} = skj = 1$ for some k. From the defn of the Boolean product, this means that

$$M_{S \circ R} = M_R \odot M_S.$$

Example 9.3.5. Find the matrix representing the relations $S \circ R$, where the matrices representing R and S are

$$M_R = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad M_S = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}.$$

Proof. The matrix for $S \circ R$ is

$$M_{S \circ R} = M_R \odot M_S = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}.$$

□

Example 9.3.6. Find the matrix representing the relations R^2, where the matrix representing R is

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

Proof. The matrix for R^2 is

$$M_{R^2} = M_R^{[2]} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

□

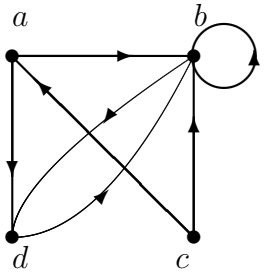
9.3.5 Representing Relations Using Digraphs

We have shown that a relation can be represented by listing all of its ordered pairs or by using a zero-one matrix. There is another important way of representing a relation using a pictorial representation. Each element of the set is represented by a point, and each ordered pair is represented using an arc with its direction indicated by an arrow. We use such pictorial representations when we think of relations on a finite set as directed graphs, or digraphs.

Definition 9.3.7. A directed graph, or digraph, consists of a set V of vertices (or nodes) together with a set E of ordered pairs of elements of V called edges (or arcs). The vertex a is called the initial vertex of the edge (a, b), and the vertex b is called the terminal vertex of this edge.

Remark 9.3.8. An edge of the form (a, a) is represented using an arc from the vertex a back to itself. Such an edge is called a loop.

Example 9.3.9. The directed graph with vertices a, b, c, and d, and edges $(a, b), (a, d), (b, b), (b, d), (c, a), (c,$ and (d, b) is displayed in the following figure.



9.3.6 Directed Graphs and Relations

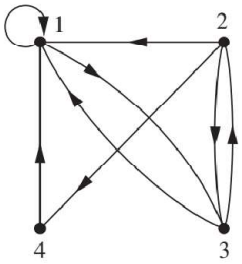
The relation R on a set A is represented by the directed graph that has the elements of A as its vertices and the ordered pairs (a, b), where $(a, b) \in R$, as edges. This assignment sets up a one-to- one correspondence between the relations on a set A and the directed graphs with A as their set of vertices. Thus, every statement about relations corresponds to a statement about directed graphs, and vice versa. Directed graphs give a visual display of information about relations. As such, they are often used to study relations and their properties. (Note that relations from a set A to a set B can be represented by a directed graph where there is a vertex for each element of A and a vertex for each element of B, as shown in Section 9.1.)

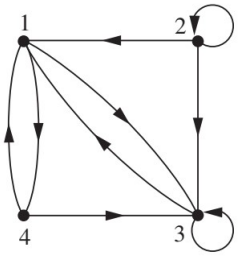
Example 9.3.10. The directed graph of the relation

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

on the set $A = \{1, 2, 3, 4\}$ is shown in the figure below.

Example 9.3.11. What are the ordered pairs in the relation R represented by the directed graph shown in the figure?





Proof. The ordered pairs (x, y) in the relation are

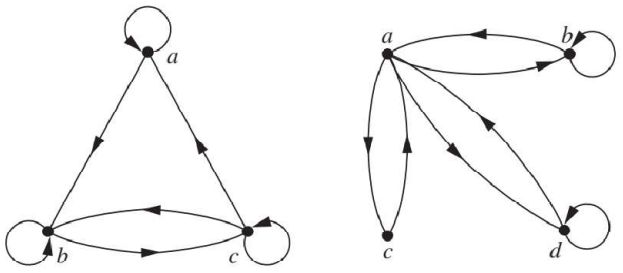
$$R = \{(1,3), (1,4), (2,1), (2,2), (2,3), (3,1), (3,3), (4,1), (4,3)\}.$$

Each of these pairs corresponds to an edge of the directed graph, with $(2,2)$ and $(3,3)$ corresponding to loops. □

9.3.7 Directed Graphs and Properties of Relations

The directed graph representing a relation can be used to determine whether the relation has various properties. For instance, a relation is reflexive if and only if there is a loop at every vertex of the directed graph, so that every ordered pair of the form (x, x) occurs in the relation. A relation is symmetric if and only if for every edge between distinct vertices in its digraph there is an edge in the opposite direction, so that (y, x) is in the relation whenever (x, y) is in the relation. Similarly, a relation is antisymmetric if and only if there are never two edges in opposite directions between distinct vertices. Finally, a relation is transitive if and only if whenever there is an edge from a vertex x to a vertex y and an edge from a vertex y to a vertex z, there is an edge from x to z (completing a triangle where each side is a directed edge with the correct direction).

Example 9.3.12. Determine whether the relations for the directed graphs shown in Figure 6 are reflexive, symmetric, antisymmetric, and/or transitive.



(a) Directed graph of R (b) Directed graph of S

Proof. Because there are loops at every vertex of the directed graph of R, it is reflexive. R is neither symmetric nor antisymmetric because there is an edge from a to b but not one from b to a, but there are edges in both directions connecting b and c. Finally, R is not transitive because there is an edge from a to b and an edge from b to c, but no edge from a to c.

Because loops are not present at all the vertices of the directed graph of S, this relation is not reflexive. It is symmetric and not antisymmetric, because every edge between distinct vertices is accompanied by an edge in the opposite direction. It is also not hard to see from the directed graph that S is not transitive, because (c, a) and (a, b) belong to S, but (c, b) does not belong to S. □

Exercises

Page 586: 1-4, 9-15, 18-28

Chapter 10

Graphs

10.1 Graphs and Graph Models

Definition 10.1.1. (Graphs)

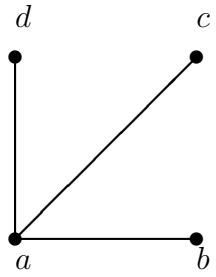
(1) A graph $G = (V, E)$ consists of V, a nonempty set of vertices (or nodes) and E, a set of edges.

(2) Each edge has either one or two vertices associated with it, called its endpoints.

(3) An edge is said to connect its endpoints.

Notation 10.1.2. If v and w are different vertices in V and e is an edge with endpoints v and w, then we write $e = \{v, w\}$ or $e = vw = wv$.

Example 10.1.3. Consider the graph $G = (V, E)$ with vertices $V = \{a, b, c, d\}$ and edges $E = \{ab, ac, ad\}$. Usually we draw a picture of a graph rather than presenting it formally as sets of vertices and edges. This graph can be described as shown in the following figure.



Remarks 10.1.4.

(1) The set of vertices V of a graph G may be infinite.

(2) A graph with an infinite vertex set or an infinite number of edges is called an infinite graph, and in comparison, a graph with a finite vertex set and a finite edge set is called a finite graph.

(3) In this course we will usually consider only finite graphs.

Definition 10.1.5.

(1) A graph in which each edge connects two different vertices and where no two edges connect the same pair of vertices is called a simple graph.

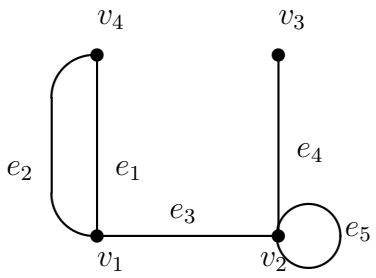
(2) Graphs that may have multiple edges connecting the same vertices are called multigraphs.

(3) When there are m different edges associated to the same unordered pair of vertices $\{u, v\}$, we also say that $\{u, v\}$ is an edge of multiplicity m. That is, we can think of this set of edges as m different copies of an edge $\{u, v\}$.

(4) Edges that connect a vertex to itself are called loops.

(5) Graphs that may include loops, and possibly multiple edges connecting the same pair of vertices or a vertex to itself, are sometimes called pseudographs.

Example 10.1.6. The following graph $G = (V, E)$ is a pseudograph with four vertices and five edges.



Directed Graphs

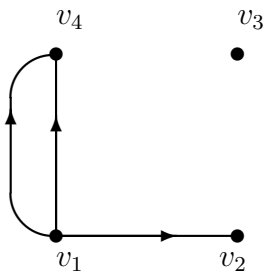
Definition 10.1.7. (Directed Graphs)

(1) A directed graph (or digraph) (V, E) consists of a nonempty set of vertices V and a set of directed edges (or arcs) E.

(2) Each directed edge is associated with an ordered pair of vertices.

(3) The directed edge associated with the ordered pair (u, v) is said to start at u and end at v.

Example 10.1.8. The following graph $G = (V, E)$ is a directed graph.



10.1.1 Types of Graphs

(1) When a directed graph has no loops and has no multiple directed edges, it is called a simple directed graph. Because a simple directed graph has at most one edge associated to each ordered pair of vertices (u, v), we call (u, v) an edge if there is an edge associated to it in the graph.

(2) Directed graphs that may have multiple directed edges from a vertex to a second (possibly the same) vertex are called directed multigraphs.

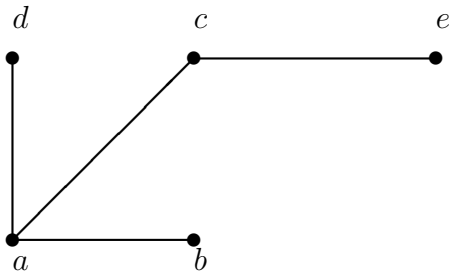
(3) When there are m directed edges, each associated to an ordered pair of vertices (u, v), we say that (u, v) is an edge of multiplicity m.

(4) A graph with both directed and undirected edges is called a mixed graph.

TABLE 1: Graph Terminology

Type	Edges	Multiple Edges Allowed?	Loops Allowed?
Simple graph	Undirected	No	No
Multigraph	Undirected	Yes	No
Pseudograph	Undirected	Yes	Yes
Simple directed graph	Directed	No	No
Directed multigraph	Directed	Yes	Yes
Mixed graph	Directed and undirected	Yes	Yes

Example 10.1.9. Determine whether the graph has directed or undirected edges, whether it has multiple edges, and whether it has one or more loops. Use your answers to determine



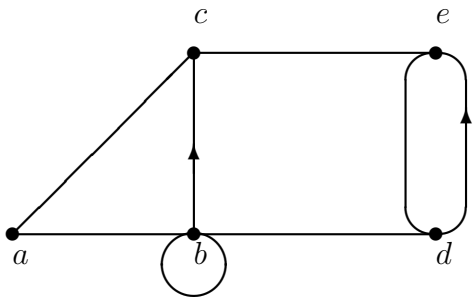
the type of graph.

Proof.

- Edges: undirected.
- Multiple Edges Allowed? No
- Loops Allowed? No
- Type: It is a simple graph.

□

Example 10.1.10. Determine whether the graph has directed or undirected edges, whether it has multiple edges, and whether it has one or more loops. Use your answers to determine



the type of graph.

Proof.

- Edges: directed and undirected
- Multiple Edges Allowed? Yes
- Loops Allowed? Yes
- Type: It is a mixed graph.

□

Exercises

Page 549: 3-10, 13

10.2 Graph Terminology and Special Types of Graphs

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

10.2.1 Basic Terminology

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

Definition 10.2.1. (Adjacent vertices)

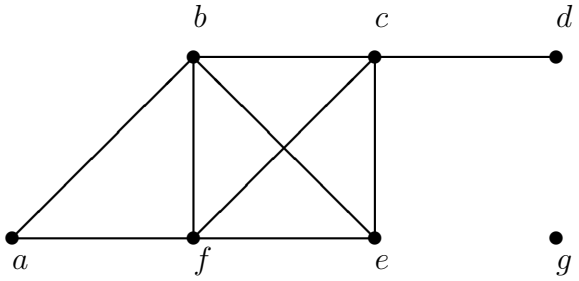
(1) Two vertices u and v in an undirected graph G are called adjacent (or neighbors) in G if u and v are endpoints of an edge e of G. Such an edge e is called incident with the vertices u and v and e is said to connect u and v.

(2) The set of all neighbors of a vertex v of $G = (V, E)$, denoted by $N(v)$, is called the neighborhood of v. If A is a subset of V, we denote by $N(A)$ the set of all vertices in G that are adjacent to at least one vertex in A. So,

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

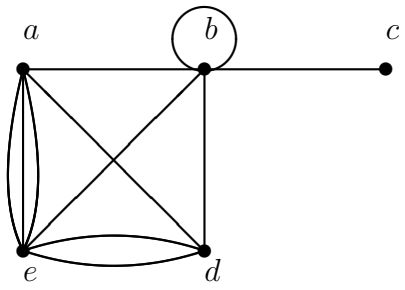
(3) The degree of a vertex in an undirected graph is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex. The degree of the vertex v is denoted by $\deg(v)$.

Example 10.2.2. What are the degrees and what are the neighborhoods of the vertices in the graph G in the given Figure?



Proof. In G, $\deg(a) = 2$, $\deg(b) = deg(c) = deg(f) = 4$, $\deg(d) = 1$, $\deg(e) = 3$, and $\deg(g) = 0$. The neighborhoods of these vertices are $N(a) = \{b, f\}$, $N(b) = \{a, c, e, f\}$, $N(c) = \{b, d, e, f\}$, $N(d) = \{c\}$, $N(e) = \{b, c, f\}$, $N(f) = \{a, b, c, e\}$, and $N(g) = \emptyset$ □

Example 10.2.3. What are the degrees and what are the neighborhoods of the vertices in the graph H in the given Figure?



Proof. In H, $\deg(a) = 5$, $\deg(b) = deg(e) = 6$, $\deg(c) = 1$, and $\deg(d) = 4$. The neighborhoods of these vertices are $N(a) = \{b, d, e\}$, $N(b) = \{a, b, c, d, e\}$, $N(c) = \{b\}$, $N(d) = \{a, b, e\}$, and $N(e) = \{a, b, d\}$. □

Definition 10.2.4.

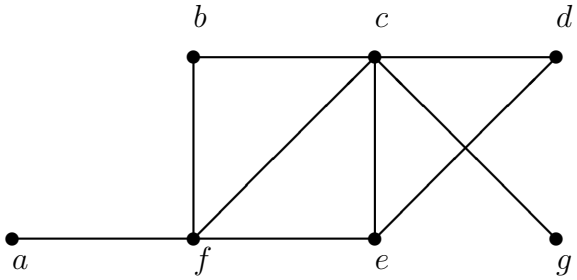
(1) A vertex of degree zero is called isolated.

(2) A vertex is pendant if it has degree one.

Theorem 10.2.5 (The Handshaking Theorem)**.** *Let $G = (V, E)$ be an undirected graph with m edges. Then*

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

Example 10.2.6. Find the sum of the degrees of the vertices of the following graph and verify that it equals twice the number of edges in the graph.



Proof. The number of edges is $m = 9$. $\deg(a) = 1$, $\deg(b) = 2$, $\deg(c) = 5$, $\deg(d) = 2$, $\deg(e) = 3$, $\deg(f) = 4$, $\deg(g) = 1$. Thus

$$2m = 18 = \sum_{v \in V} \deg(v).$$

□

Theorem 10.2.7. *An undirected graph has an even number of vertices of odd degree.*

Proof. Let V_1 and V_2 be the set of vertices of even degree and the set of vertices of odd degree, respectively, in an undirected graph $G = (V, E)$ with m edges. Then

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

Because $\deg(v)$ is even for $v \in V_1$, $\sum_{v \in V_1} \deg(v)$ is even. Hence, $\sum_{v \in V_2} \deg(v) = 2m - \sum_{v \in V_1} \deg(v)$ is also even. Because all the terms in $\sum_{v \in V_2} \deg(v)$ are odd, there must be an even number of such terms. Thus, there are an even number of vertices of odd degree. □

10.2.2 Basic Terminology for Digraphs

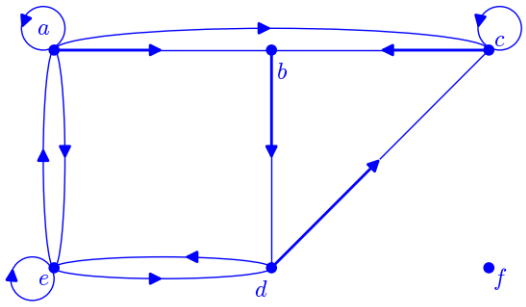
Terminology for graphs with directed edges reflects the fact that edges in directed graphs have directions.

Definition 10.2.8. (Adjacent to and adjacent from)
When (u, v) is an edge of the graph G with directed edges, u is said to be adjacent to v and v is said to be adjacent from u. The vertex u is called the initial vertex of (u, v), and v is called the terminal or end vertex of (u, v). The initial vertex and terminal vertex of a loop are the same.

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

Definition 10.2.9. (In-degree and out-degree)
In a graph with directed edges the in-degree of a vertex v, denoted by $\deg^-(v)$, is the number of edges with v as their terminal vertex. The out-degree of v, denoted by $\deg^+(v)$, is the number of edges with v as their initial vertex. (Note that a loop at a vertex contributes 1 to both the in-degree and the out-degree of this vertex.)

Example 10.2.10. Find the in-degree and out-degree of each vertex in the graph G with directed edges shown in the figure.



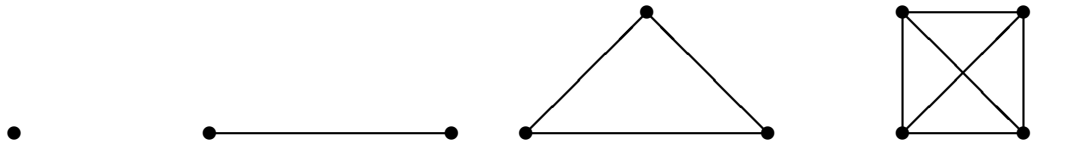
Proof. The in-degrees in G are $\deg^-(a) = 2$, $\deg^-(b) = 2$, $\deg^-(c) = 3$, $\deg^-(d) = 2$, $\deg^-(e) = 3$, and $\deg^-(f) = 0$. The out-degrees are $\deg^+(a) = 4$, $\deg^+(b) = 1$, $\deg^+(c) = 2$, $\deg^+(d) = 2$, $\deg^+(e) = 3$, and $\deg^+(f) = 0$. □

10.2.3 Some Special Simple Graphs

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

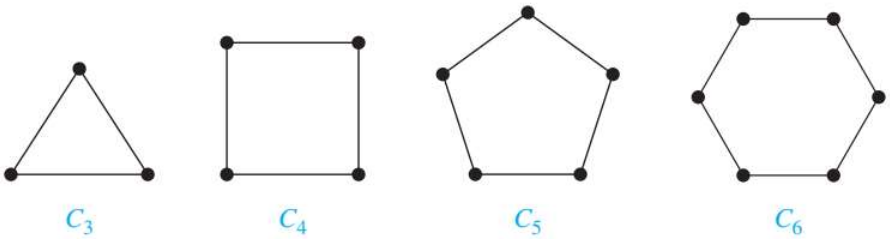
Definition 10.2.11. (Complete Graphs)
A complete graph on n vertices, denoted by K_n, is a simple graph that contains exactly one edge between each pair of distinct vertices.

Example 10.2.12. The graphs K_n, for $n = 1, 2, 3, 4$, are displayed in the following figure.



Definition 10.2.13. (Cycles)
A cycle C_n, $n \geq 3$, consists of n vertices $v_1, v_2, \cdots, v_n$ and edges $\{v_1, v_2\}, \{v_2, v_3\}, \cdots, \{v_{n-1}, v_n\}$, and $\{v_n, v_1\}$.

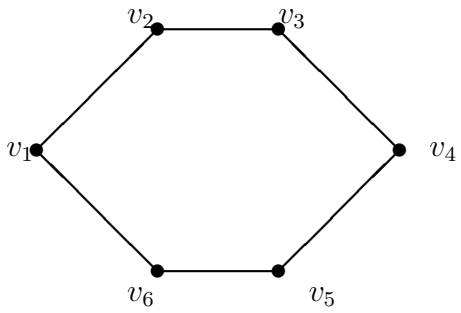
Example 10.2.14. The graphs C_n, for $n = 3, 4, 5, 6$, are displayed in the following figure.


C_3
C_4
C_5
C_6

10.2.4 Bipartite Graphs

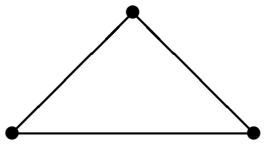
Definition 10.2.15. (Bipartite Graphs)
A simple graph G is called bipartite if its vertex set V can be partitioned into two disjoint sets V_1 and V_2 such that every edge in the graph connects a vertex in V_1 and a vertex in V_2 (so that no edge in G connects either two vertices in V_1 or two vertices in V_2). When this condition holds, we call the pair (V_1, V_2) a bipartition of the vertex set V of G.

Example 10.2.16. Determine whether the cycle C_6 is bipartite.



Proof. The cycle C_6 is bipartite, because its vertex set can be partitioned into the two sets $V_1 = \{v_1, v_3, v_5\}$ and $V_2 = \{v_2, v_4, v_6\}$, and every edge of G connects a vertex in V_1 and a vertex in V_2. □

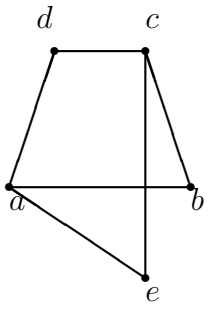
Example 10.2.17. Show that complete graph K_3 is not bipartite.



Proof. The graph is not bipartite, because if we divide the vertex set of K_3 into two disjoint sets, one of the two sets must contain two vertices. If the graph were bipartite, these two vertices could not be connected by an edge, but in K_3 each vertex is connected to every other vertex by an edge. □

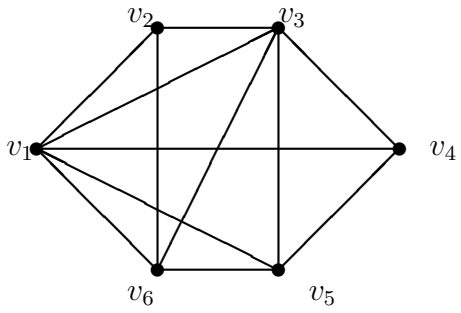
Theorem 10.2.18. *A simple graph is bipartite if and only if it is possible to assign one of two different colors to each vertex of the graph so that no two adjacent vertices are assigned the same color.*

Example 10.2.19. Determine whether the graph G is bipartite.



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

Example 10.2.20. Determine whether the graph G is bipartite.



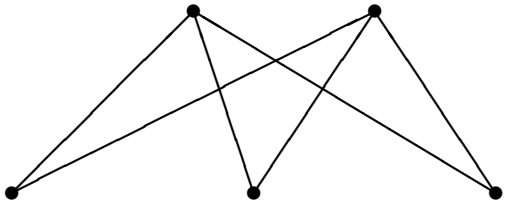
Proof. Without loss of generality we arbitrarily assign red to v_2. Then, we must assign blue to v_6, v_1, and v_3, because each is adjacent to v_2. But this is not possible because v_6 and v_1 are adjacent, so both cannot be assigned blue. This argument shows that we cannot assign one of two colors to each of the vertices of G so that no adjacent vertices are assigned the same color. It follows by Theorem 4 that G is not bipartite. □

10.2.5 Complete Bipartite Graphs

- A complete bipartite graph $K_{m,n}$ is a graph that has its vertex set partitioned into two subsets of m and n vertices, respectively with an edge between two vertices if and only if one vertex is in the first subset and the other vertex is in the second subset.

Example 10.2.21. Graph $K_{2,3}$.

Proof.



□

10.2.6 New Graphs from Old

Sometimes we need only part of a graph to solve a problem. When edges and vertices are removed from a graph, without removing endpoints of any remaining edges, a smaller graph is obtained. Such a graph is called a subgraph of the original graph.

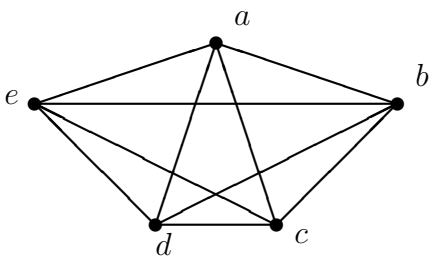
Definition 10.2.22. (Subgraphs)

(1) A subgraph of a graph $G = (V, E)$ is a graph $H = (W, F)$, where $W \subset V$ and $F \subset E$.

(2) A subgraph H of G is a proper subgraph of G if $H \neq G$.

(3) Let $G = (V, E)$ be a simple graph. The subgraph induced by a subset W of the vertex set V is the graph (W, F), where the edge set F contains an edge in E if and only if both endpoints of this edge are in W.

Example 10.2.23. Consider the complete graph K_5.



The graph $G = \{\{a, b, c, e\}, \{ab, ac, ae, bc, be\}\}$ is a subgraph of K_5. If we add the edge connecting c and e to G, we obtain the subgraph induced by $W = \{a, b, c, e\}$.

Exercises

Page 665: 1-5, 7-10, 21-25, 51

10.3 Representing Graphs and Graph Isomorphism

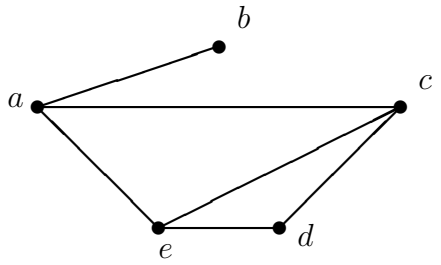
There are many useful ways to represent graphs. As we will see throughout this chapter, in working with a graph it is helpful to be able to choose its most convenient representation. In this section we will show how to represent graphs in several different ways.

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

10.3.1 Representing Graphs

One way to represent a graph without multiple edges is to list all the edges of this graph. Another way to represent a graph with no multiple edges is to use adjacency lists, which specify the vertices that are adjacent to each vertex of the graph.

Example 10.3.1. Use adjacency lists to describe the simple given graph.



Proof. The following table lists those vertices adjacent to each of the vertices of the graph.

An Adjacency List for a Simple Graph	
Vertex	Adjacent Vertices
a	b , c , e
b	a
c	a , d , e
d	c , e
e	a , c , d

□

10.3.2 Adjacency Matrices

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

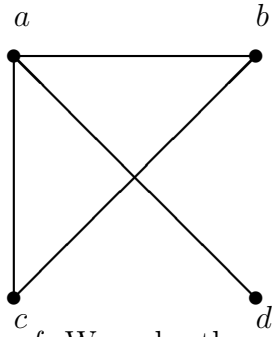
Definition 10.3.2. Suppose that $G = (V, E)$ is a simple graph where $|V| = n$. Suppose that the vertices of G are listed arbitrarily as $v_1, v_2, \cdots, v_n$. The adjacency matrix A (or A_G) of G, with respect to this listing of the vertices, is the $n \times n$ zero-one matrix with 1 as its (i, j)th entry when v_i and v_j are adjacent, and 0 as its (i, j)th entry when they are not adjacent. In other words, if its adjacency matrix is $A = [a_{ij}]$, then

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

Note that an adjacency matrix of a graph is based on the ordering chosen for the vertices. Hence, there may be as many as $n!$ different adjacency matrices for a graph with n vertices, because there are $n!$ different orderings of n vertices.

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

Example 10.3.3. Use an adjacency matrix to represent the following graph.



Proof. We order the vertices as a, b, c, d. The matrix representing this graph is

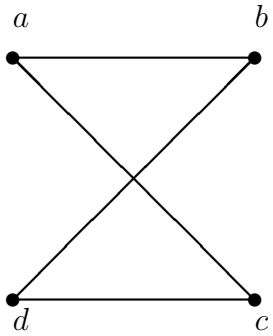
$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

□

Example 10.3.4. Draw a graph with the adjacency matrix

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

Proof. A graph with this adjacency matrix is

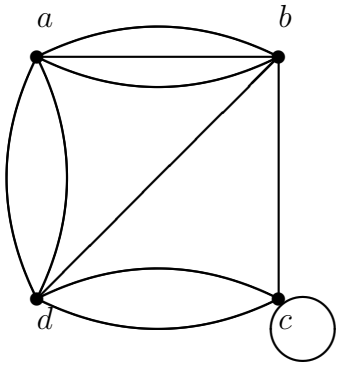


□

10.3.3 Adjacency Matrices For Multigraphs

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

Example 10.3.5. Use an adjacency matrix to represent the following graph.



Proof. We order the vertices as a, b, c, d. The matrix representing this graph is

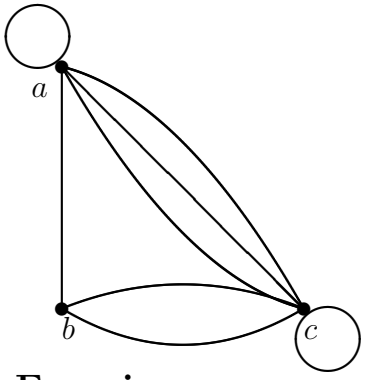
$$\begin{bmatrix} 0 & 3 & 0 & 2 \\ 3 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 \\ 2 & 1 & 2 & 0 \end{bmatrix}.$$

□

Example 10.3.6. Draw a graph with the adjacency matrix

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

Proof. A graph with this adjacency matrix is



□

Exercises

Page 675: 1,2,5,6,9-18, 22-24,

10.4 Connectivity

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

10.4.1 Paths

Informally, a path is a sequence of edges that begins at a vertex of a graph and travels from vertex to vertex along edges of the graph. As the path travels along its edges, it visits the vertices along this path, that is, the endpoints of these edges.

Definition 10.4.1.

(1) Let n be a nonnegative integer and G an undirected graph. A path of length n from u to v in G is a sequence of n edges $e_1, \cdots, e_n$ of G for which there exists a sequence $x_0 = u, x_1, \cdots, x_{n-1}, x_n = v$ of vertices such that e_i has, for $i = 1, \cdots, n$, the endpoints x_{i-1} and x_i.

(2) When the graph is simple, we denote this path by its vertex sequence $x_0, x_1, \cdots, x_{n-1}, x_n$ (because listing these vertices uniquely determines the path).

(3) The path is a circuit if it begins and ends at the same vertex, that is, if $u = v$, and has length greater than zero.

(4) The path or circuit is said to pass through the vertices $x_1, x_2, \cdots, x_{n-1}$ or traverse the edges $e_1, \cdots, e_n$.

(5) A path or circuit is simple if it does not contain the same edge more than once.

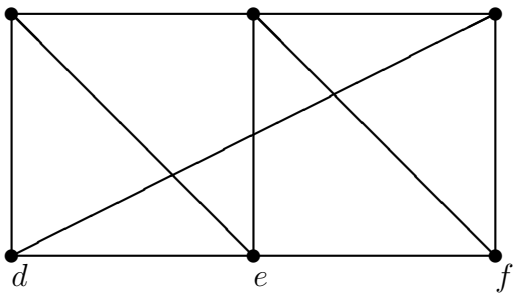
Remarks 10.4.2.

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

(2) A path of length zero consists of a single vertex.

(3) In some books, the term walk is used instead of path, where a walk is defined to be an alternating sequence of vertices and edges of a graph, $v_0, e_1, v_1, e_2, \cdots, v_{n-1}, e_n, v_n$, where v_{i-1} and v_i are the endpoints of e_i for $i = 1, 2, \cdots, n$. When this terminology is used, closed walk is used instead of circuit, and trail is used instead of the term simple path. When this terminology is used, the terminology path is often used for a trail with no repeated vertices.

Example 10.4.3. Consider the simple graph. a, d, c, f, e is a simple path of length 4,



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

Definition 10.4.4.

(1) Let n be a nonnegative integer and G a directed graph. A path of length n from u to v in G is a sequence of n edges $e_1, \cdots, e_n$ of G such that e_1 is associated with (x_0, x_1), e_2 is associated with (x_1, x_2), and so on, with e_n associated with (x_{n-1}, x_n) where $x_0 = u, x_1, \cdots, x_{n-1}, x_n = v$.

(2) When there are no multiple edges in the directed graph, this path is denoted by its vertex sequence $x_0, x_1, \cdots, x_{n-1}, x_n$.

(3) A path of length greater than zero that begins and ends at the same vertex is called a circuit or cycle.

(4) A path or circuit is called simple if it does not contain the same edge more than once.

10.4.2 Connectedness in Undirected Graphs

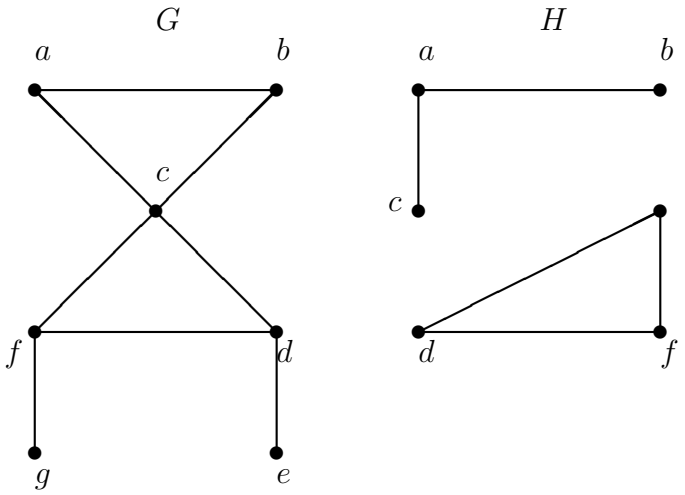
Definition 10.4.5.

(1) An undirected graph is called connected if there is a path between every pair of distinct vertices of the graph.

(2) An undirected graph that is not connected is called disconnected.

(3) We say that we disconnect a graph when we remove vertices or edges, or both, to produce a disconnected subgraph.

Example 10.4.6. The graph G in the figure is connected, because for every pair of distinct vertices there is a path between them (the reader should verify this). However, the graph H in the figure is not connected. For instance, there is no path in H between vertices a and d.



10.4.3 Paths and Isomorphism

There are several ways that paths and circuits can help determine whether two graphs are isomorphic. For example, the existence of a simple circuit of a particular length is a useful invariant that can be used to show that two graphs are not isomorphic. In addition, paths can be used to construct mappings that may be isomorphisms.

Exercises

Page 689: 1-5, 20-23

10.5 Euler and Hamilton Paths

Can we travel along the edges of a graph starting at a vertex and returning to it by traversing each edge of the graph exactly once? Similarly, can we travel along the edges of a graph starting at a vertex and returning to it while visiting each vertex of the graph exactly once? Although these questions seem to be similar, the first question, which asks whether a graph has an Euler circuit, can be easily answered simply by examining the degrees of the vertices of the graph, while the second question, which asks whether a graph has a Hamilton circuit, is quite difficult to solve for most graphs. In this section we will study these questions and discuss the difficulty of solving them. Although both questions have many practical applications in many different areas, both arose in old puzzles.

10.5.1 Euler Paths and Circuits

The town of Königsberg, Prussia (now called Kaliningrad and part of the Russian republic), was divided into four sections by the branches of the Pregel River. These four sections included the two regions on the banks of the Pregel, Kneiphof Island, and the region between the two branches of the Pregel. In the eighteenth century seven bridges connected these regions. Figure 1 depicts these regions and bridges.

The townspeople took long walks through town on Sundays. They wondered whether it was possible to start at some location in the town, travel across all the bridges once without crossing any bridge twice, and return to the starting point.

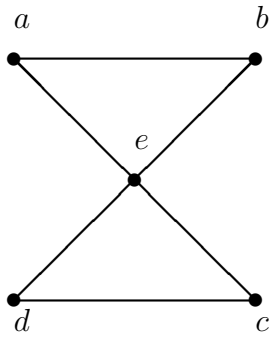
The Swiss mathematician Leonhard Euler solved this problem. His solution, published in 1736, may be the first use of graph theory. Euler studied this problem using the multigraph obtained when the four regions are represented by vertices and the bridges by edges.

The problem of traveling across every bridge without crossing any bridge more than once can be rephrased in terms of this model. The question becomes: Is there a simple circuit in this multigraph that contains every edge?

Definition 10.5.1. (1) An Euler circuit in a graph G is a simple circuit containing every edge of G.

(2) An Euler path in G is a simple path containing every edge of G.

Example 10.5.2. Determine whether the given graph has an Euler circuit. Construct such a circuit when one exists. If no Euler circuit exists, determine whether the graph has an Euler path and construct such a path if one exists.



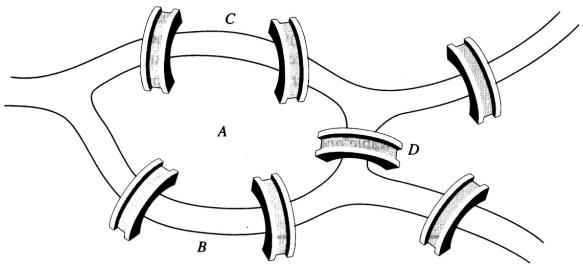
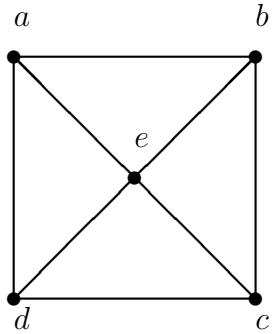


Figure 10.1: The Seven Bridges of Königsberg

Proof. The graph has an Euler circuit, for example, a, e, c, d, e, b, a. □

Example 10.5.3. Determine whether the given graph has an Euler circuit. Construct such a circuit when one exists. If no Euler circuit exists, determine whether the graph has an Euler path and construct such a path if one exists.



Proof. The graphs does not has an Euler circuit and it does not have an Euler path. □

Example 10.5.4. Determine whether the given graph has an Euler circuit. Construct such a circuit when one exists. If no Euler circuit exists, determine whether the graph has an

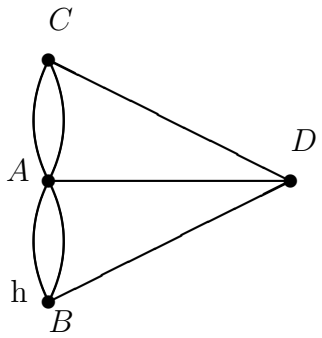
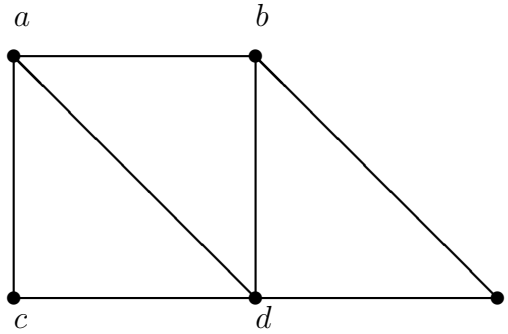


Figure 10.2: Multigraph Model of the Town of Königsberg.

Euler path and construct such a path if one exists.



Proof. The graphs does not has an Euler circuit but it has an Euler path namely, a, c, d, e, b, d, a, b. □

10.5.2 Necessary And Sufficient Conditions For Euler Circuits And Paths

here are simple criteria for determining whether a multigraph has an Euler circuit or an Euler path. Euler discovered them when he solved the famous Königsberg bridge problem. We will assume that all graphs discussed in this section have a finite number of vertices and edges. What can we say if a connected multigraph has an Euler circuit? What we can show is that every vertex must have even degree. To do this, first note that an Euler circuit begins with a vertex a and continues with an edge incident with a, say $\{a, b\}$. The edge $\{a, b\}$ contributes one to $\deg(a)$. Each time the circuit passes through a vertex it contributes two to the vertex's degree, because the circuit enters via an edge incident with this vertex and leaves via another such edge. Finally, the circuit terminates where it started, contributing one to $\deg(a)$. Therefore, $\deg(a)$ must be even, because the circuit contributes one when it begins, one when it ends, and two every time it passes through a (if it ever does). A vertex other than a has even degree because the circuit contributes two to its degree each time it passes through the vertex. We conclude that if a connected graph has an Euler circuit, then every vertex must have even degree.

Is this necessary condition for the existence of an Euler circuit also sufficient? That is, must an Euler circuit exist in a connected multigraph if all vertices have even degree? This

question can be settled affirmatively with a construction.

Suppose that G is a connected multigraph with at least two vertices and the degree of every vertex of G is even. We will form a simple circuit that begins at an arbitrary vertex a of G, building it edge by edge. Let $x_0 = a$. First, we arbitrarily choose an edge $\{x_0, x_1\}$ incident with a which is possible because G is connected. We continue by building a simple path $\{x_0, x_1\}, \{x_1, x_2\}, \cdots, \{x_{n-1}, x_n\}$, successively adding edges one by one to the path until we cannot add another edge to the path. This happens when we reach a vertex for which we have already included all edges incident with that vertex in the path.

The path we have constructed must terminate because the graph has a finite number of edges, so we are guaranteed to eventually reach a vertex for which no edges are available to add to the path. The path begins at a with an edge of the form $\{a, x\}$, and we now show that it must terminate at a with an edge of the form $\{y, a\}$. To see that the path must terminate at a, note that each time the path goes through a vertex with even degree, it uses only one edge to enter this vertex, so because the degree must be at least two, at least one edge remains for the path to leave the vertex. Furthermore, every time we enter and leave a vertex of even degree, there are an even number of edges incident with this vertex that we have not yet used in our path. Consequently, as we form the path, every time we enter a vertex other than a, we can leave it. This means that the path can end only at a. Next, note that the path we have constructed may use all the edges of the graph, or it may not if we have returned to a for the last time before using all the edges.

Theorem 10.5.5. *A connected multigraph with at least two vertices has an Euler circuit if and only if each of its vertices has even degree.*

Proof. □

Theorem 10.5.6. *A connected multigraph has an Euler path but not an Euler circuit if and only if it has exactly two vertices of odd degree.*

Proof. □

10.5.3 Hamilton Paths and Circuits

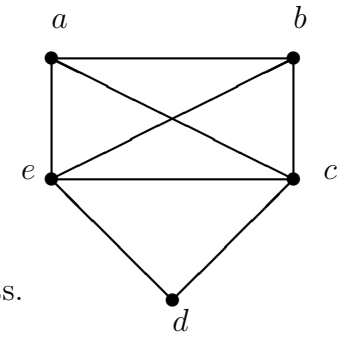
We have developed necessary and sufficient conditions for the existence of paths and circuits that contain every edge of a multigraph exactly once. Can we do the same for simple paths and circuits that contain every vertex of the graph exactly once?

Definition 10.5.7. A simple path in a graph G that passes through every vertex exactly once is called a Hamilton path, and a simple circuit in a graph G that passes through every vertex exactly once is called a Hamilton circuit. That is, the simple path $x_0, x_1, \cdots, x_{n-1}, x_n$ in the graph $G = (V, E)$ is a Hamilton path if $V = \{x_0, x_1, \cdots, x_{n-1}, x_n\}$ and $x_i \neq x_j$ for $0 \leq i < j \leq n$, and the simple circuit x $x_0, x_1, \cdots, x_{n-1}, x_n m x_0$ (with $n > 0$) is a Hamilton circuit if $x_0, x_1, \cdots, x_{n-1}, x_n$ is a Hamilton path.

This terminology comes from a game, called the Icosian puzzle, invented in 1857 by the Irish mathematician Sir William Rowan Hamilton. It consisted of a wooden dodecahedron,

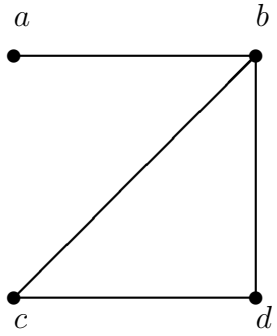
with a peg at each vertex of the dodecahedron, and string. The 20 vertices of the dodecahedron were labeled with different cities in the world. The object of the puzzle was to start at a city and travel along the edges of the dodecahedron, visiting each of the other 19 cities exactly once, and end back at the first city. The circuit traveled was marked off using the string and pegs.

Example 10.5.8. Determine whether the given graph has a Hamilton circuit. Construct such a circuit when one exists. If no Hamilton circuit exists, determine whether the graph has a Hamilton path and construct such a path if one exists.



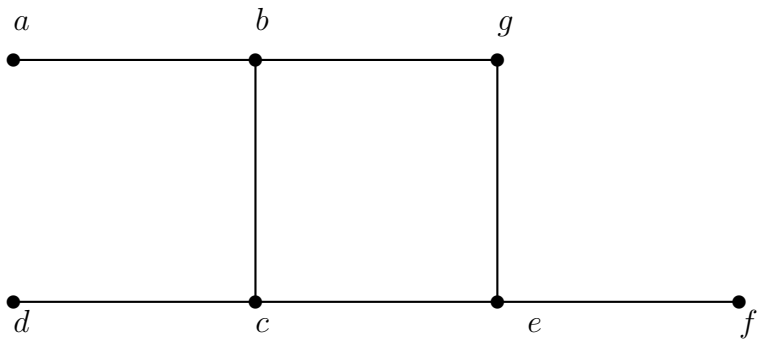
Proof. The graph has has a Hamilton circuit: a, b, c, d, e, a. □

Example 10.5.9. Determine whether the given graph has a Hamilton. Construct such a circuit when one exists. If no Hamilton circuit exists, determine whether the graph has a Hamilton path and construct such a path if one exists.



Proof. There is no Hamilton circuit in the graphs (this can be seen by noting that any circuit containing every vertex must contain the edge $\{a, b\}$ twice), but it does have a Hamilton path, namely, a, b, c, d. □

Example 10.5.10. Determine whether the given graph has an Euler circuit. Construct such a circuit when one exists. If no Euler circuit exists, determine whether the graph has an Euler path and construct such a path if one exists.



Proof. The graphs has neither a Hamilton circuit nor a Hamilton path, because any path containing all vertices must contain one of the edges $\{a, b\}$, $\{e, f\}$, and $\{c, d\}$ more than once. □

10.5.4 Conditions For The Existence Of Hamilton Circuits

Is there a simple way to determine whether a graph has a Hamilton circuit or path?At first, it might seem that there should be an easy way to determine this, because there is a simple way to answer the similar question of whether a graph has an Euler circuit. Surprisingly, there are no known simple necessary and sufficient criteria for the existence of Hamilton circuits. However, many theorems are known that give sufficient conditions for the existence of Hamilton circuits. Also, certain properties can be used to show that a graph has no Hamilton circuit. For instance, a graph with a vertex of degree one cannot have a Hamilton circuit, because in a Hamilton circuit, each vertex is incident with two edges in the circuit. Moreover, if a vertex in the graph has degree two, then both edges that are incident with this vertex must be part of any Hamilton circuit. Also, note that when a Hamilton circuit is being constructed and this circuit has passed through a vertex, then all remaining edges incident with this vertex, other than the two used in the circuit, can be removed from consideration. Furthermore, a Hamilton circuit cannot contain a smaller circuit within it.

Theorem 10.5.11 (DIRAC'S THEOREM)**.** *If G is a simple graph with n vertices with $n \geq 3$ such that the degree of every vertex in G is at least $n/2$, then G has a Hamilton circuit.*

Theorem 10.5.12 (ORE'S THEOREM)**.** *If G is a simple graph with n vertices with $n \geq 3$ such that $\deg(u) + \deg(v) \geq n$ for every pair of nonadjacent vertices u and v in G, then G has a Hamilton circuit.*

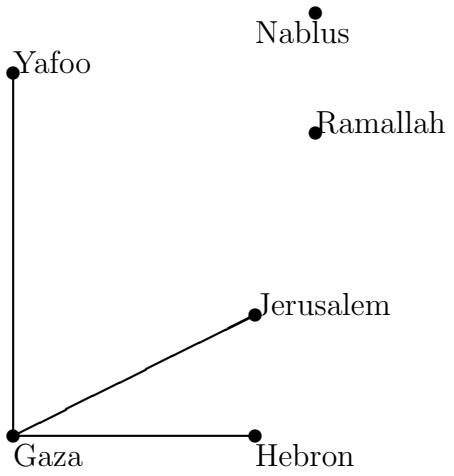
Both Ore's theorem and Dirac's theorem provide sufficient conditions for a connected simple graph to have a Hamilton circuit. However, these theorems do not provide necessary conditions for the existence of a Hamilton circuit. For example, the graph C 5 has a Hamilton circuit but does not satisfy the hypotheses of either Ore's theorem or Dirac's theorem, as the reader can verify.

Exercises

Page 703: 1-8, 30-40

10.6 Shortest-Path Problems

Many problems can be modeled using graphs with weights assigned to their edges. As an illustration, consider how an airline system can be modeled. We set up the basic graph model by representing cities by vertices and flights by edges. Problems involving distances can be modeled by assigning distances between cities to the edges. Problems involving flight time can be modeled by assigning flight times to edges.



Definition 10.6.1. (Weighted graphs)
Graphs that have a number assigned to each edge are called weighted graphs.

Several types of problems involving weighted graphs arise frequently. Determining a path of least length between two vertices in a network is one such problem. To be more specific, let the length of a path in a weighted graph be the sum of the weights of the edges of this path. (The reader should note that this use of the term length is different from the use of length to denote the number of edges in a path in a graph without weights.) The question is: What is a shortest path, that is, a path of least length, between two given vertices?

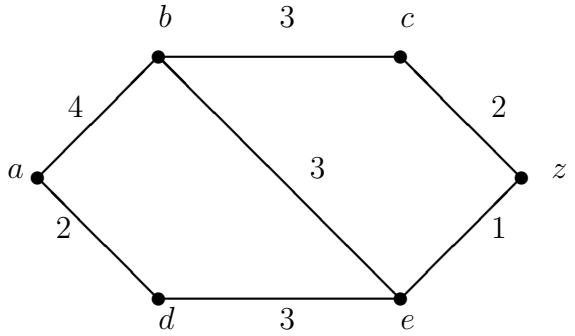
Another important problem involving weighted graphs asks for a circuit of shortest total length that visits every vertex of a complete graph exactly once. This is the famous traveling salesperson problem , which asks for an order in which a salesperson should visit each of the cities on his route exactly once so that he travels the minimum total distance. We will discuss the traveling salesperson problem later in this section.

10.6.1 A Shortest-Path Algorithm

There are several different algorithms that find a shortest path between two vertices in a weighted graph. We will present a greedy algorithm discovered by the Dutch mathematician

Edsger Dijkstra in 1959. The version we will describe solves this problem in undirected weighted graphs where all the weights are positive. It is easy to adapt it to solve shortest-path problems in directed graphs.

Example 10.6.2. What is the length of a shortest path between a and z in the weighted graph shown in the figure below?



Proof. Although a shortest path is easily found by inspection, we will develop some ideas useful in understanding Dijkstra's algorithm. We will solve this problem by finding the length of a shortest path from a to successive vertices, until z is reached.

The only paths starting at a that contain no vertex other than a are formed by adding an edge that has a as one endpoint. These paths have only one edge. They are a, b of length 4 and a, d of length 2. It follows that d is the closest vertex to a, and the shortest path from a to d has length 2.

We can find the second closest vertex by examining all paths that begin with the shortest path from a to a vertex in the set $\{a, d\}$, followed by an edge that has one endpoint in $\{a, d\}$ and its other endpoint not in this set. There are two such paths to consider, a, d, e of length 7 and a, b of length 4. Hence, the second closest vertex to a is b and the shortest path from a to b has length 4.

To find the third closest vertex to a, we need examine only the paths that begin with the shortest path from a to a vertex in the set $\{a, d, b\}$, followed by an edge that has one endpoint in the set $\{a, d, b\}$ and its other endpoint not in this set. There are three such paths, a, b, c of length 7, a, b, e of length 7, and a, d, e of length 5. Because the shortest of these paths is a, d, e, the third closest vertex to a is e and the length of the shortest path from a to e is 5.

To find the fourth closest vertex to a, we need examine only the paths that begin with the shortest path from a to a vertex in the set $\{a, d, b, e\}$, followed by an edge that has one endpoint in the set $\{a, d, b, e\}$ and its other endpoint not in this set. There are two such paths, a, b, c of length 7 and a, d, e, z of length 6. Because the shorter of these paths is a, d, e, z , the fourth closest vertex to a is z and the length of the shortest path from a to z is 6. □

The above example illustrates the general principles used in Dijkstra's algorithm. Note that a shortest path from a to z could have been found by a brute force approach by

examining the length of every path from a to z. However, this brute force approach is impractical for humans and even for computers for graphs with a large number of edges.

10.6.2 Dijkstra's algorithm

The Dijkstra's algorithm begins by labeling a with 0 and the other vertices with ı. We use the note $L_0(a) = 0$ and $L_0(v) = \text{ı}$ for these labels before any iterations have taken place (the subscript 0 stands for the "0th" iteration). These labels are the lengths of shortest paths from a to the vertices, where the paths contain only the vertex a. (Because no path from a to a vertex different from a exists, ı is the length of a shortest path between a and this vertex.)

Dijkstra's algorithm proceeds by forming a distinguished set of vertices. Let S_k denote this set after k iterations of the labeling procedure. We begin with $S_0 = \emptyset$. The set S_k is formed from S_{k-1} by adding a vertex u not in S_{k-1} with the smallest label.

Once u is added to S_k, we update the labels of all vertices not in S_k, so that $L_k(v)$, the label of the vertex v at the kth stage, is the length of a shortest path from a to v that contains vertices only in S_k (that is, vertices that were already in the distinguished set together with u). Note that the way we choose the vertex u to add to S_k at each step is an optimal choice at each step, making this a greedy algorithm.

Let v be a vertex not in S_k. To update the label of v, note that $L_k(v)$ is the length of a shortest path from a to v containing only vertices inS_k. The updating can be carried out efficiently when this observation is used: A shortest path from a to v containing only elements of S_k is either a shortest path from a to v that contains only elements of S_{k-1} (that is, the distinguished vertices not including u), or it is a shortest path from a to u at the $(k-1)$st stage with the edge $\{u, v\}$ added. In other words,

$$L_k(a, v) = \min\{L_{k-1}(a, v), L_{k-1}(a, u) + w(u, v)\},$$

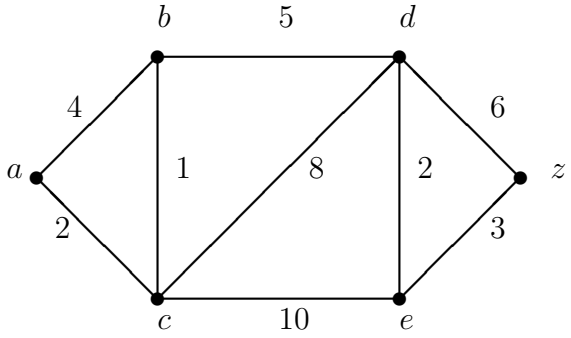
where $w(u, v)$ is the length of the edge with u and v as endpoints. This procedure is iterated by successively adding vertices to the distinguished set until z is added. When z is added to the distinguished set, its label is the length of a shortest path from a to z.

Example 10.6.3. Use Dijkstra's algorithm to find the length of a shortest path between the vertices a and z in the weighted graph displayed in the figure below?

Proof. The steps used by Dijkstra's algorithm to find a shortest path between a and z are shown in the figure. At each iteration of the algorithm the vertices of the set S_k are circled. A shortest path from a to each vertex containing only vertices in S_k is indicated for each iteration. The algorithm terminates when z is circled. We find that a shortest path from a to z is a, c, b, d, e, z, with length 13. □

Exercises

Page 716: 2-13


b
5
d
4
6
a
1
8
2
z
2
3
c
10
e