Course 2BA1: First Semester 2007–08

David R. Wilkins

Contents

1 The Principle of Mathematical Induction

1.1 Integers and Natural Numbers

An *integer* is a whole number. Such numbers are of three types, *positive*, *negative* and *zero*. The *positive integers* (or positive whole numbers) are $1, 2, 3, 4, \ldots$. Similarly the *negative integers* or negative whole numbers) are $-1, -2, -3, -4, \ldots$. There is of course exactly one integer that is zero, namely 0 itself.

The *non-negative integers* are therefore $0, 1, 2, 3, \ldots$. Similarly the *non-positive integers* are $0, -1, -2, -3, \ldots$.

It is customary in mathematics to denote the set (or collection) of integers by $\mathbb{Z}$. (The word for 'number' in German is 'Zahl'.)

The *natural numbers* are the positive integers $1, 2, 3, 4, \ldots$. It is customary to denote the set of natural numbers by $\mathbb{N}$.

(Note therefore that terms 'natural number' and 'positive integer' are synonyms, i.e., they refer to the same objects.)

1.2 Introduction to the Principle of Mathematical Induction

For each natural number n, let S_n denote the sum of the first n (positive) odd numbers. Calculating S_1, S_2, S_3, S_4, S_5, we find

$$\begin{array}{lll} S_1 & = 1 & = 1, \\ S_2 & = 1 + 3 & = 4, \\ S_3 & = 1 + 3 + 5 & = 9, \\ S_4 & = 1 + 3 + 5 + 7 & = 16, \\ S_5 & = 1 + 3 + 5 + 7 + 9 & = 25. \end{array}$$

You may notice a pattern beginning to emerge. Does this pattern continue? Suppose that we see whether or not the pattern continues to S_6. Adding up, we find

$$S_6 = 1 + 3 + 5 + 7 + 9 + 11 = 36.$$

We are thus led to conjecture that

$$S_n = n^2$$

for all natural numbers n?

Can we prove it? If so, how?

Merely testing the proposition for a few values of n, no matter how many, cannot in itself suffice to *prove* that the proposition holds for *all* natural

numbers n. Moreover propositions may turn out to be true in a very large number of cases, and yet fail for others. Such a proposition is the following:

$$\text{“}n < 1,000,000,000\text{”}.$$

This proposition holds for a large number of natural numbers n (indeed for $999,999,999$ of them, to be precise), yet it obviously fails to hold for all natural numbers n.

One might ask what strategies are available for proving that some conjectured result does indeed hold for all natural numbers n. One such is the *Principle of Mathematical Induction*.

Suppose that, for each natural number n, $P(n)$ denotes some proposition, such as “$S_n = n^2$”. For each value of n, the proposition $P(n)$ would be either true or false. Our task is to prove that it is true for all values of n. The Principle of Mathematical Induction states that this is true provided that (i) $P(1)$ is true, and (ii) if $P(m)$ is true for any natural number m then $P(m+1)$ is also true.

We can express this more informally as follows. Suppose that we are required to prove that some statement is true for all values of a natural number n. To do this, it suffices to prove (i) that the statement is true when $n = 1$, and (ii) that if the statement is true when $n = m$ for some natural number m, then it is also true when $n = m + 1$ (no matter what the value of m).

To understand the justification for the Principle of Mathematical Induction, consider the following. For each natural number n, let $P(n)$ denote (as above) a proposition (that is either true or false). We suppose that we have proved that $P(1)$ is true, and that if $P(m)$ is true then $P(m + 1)$ is true. Now

$P(1)$ is true.

If $P(1)$ is true then $P(2)$ is true. Moreover $P(1)$ is true.
Therefore $P(2)$ is true.

If $P(2)$ is true then $P(3)$ is true. Moreover $P(2)$ is true.
Therefore $P(3)$ is true.

If $P(3)$ is true then $P(4)$ is true. Moreover $P(3)$ is true.
Therefore $P(4)$ is true.

$\vdots$

If $P(n-2)$ is true then $P(n-1)$ is true. Moreover $P(n-2)$ is true. Therefore $P(n-1)$ is true.

If $P(n-1)$ is true then $P(n)$ is true. Moreover $P(n-1)$ is true.
Therefore $P(n)$ is true.

The pattern exhibited in these statements should convince you that $P(n)$ is true for any natural number n, no matter how large.

We now consider how to apply the Principle of Mathematical Induction to prove that $S_n = n^2$ for all natural numbers n, where S_n denotes the sum of the first n odd numbers. Obviously $S_1 = 1$, so that the conjectured result holds when $n = 1$. Suppose that $S_m = m^2$ for some natural number m. Then

$$S_{m+1} = S_m + (2m + 1) = m^2 + 2m + 1 = (m + 1)^2$$

Thus if the identity $S_n = n^2$ holds when $n = m$ then it also holds when $n = m + 1$. We conclude from the Principle of Mathematical Induction that $S_n = n^2$ for all natural numbers n.

We can write out the argument rather more formally as follows. For each natural number n, let $P(n)$ denote the proposition "$S_n = n^2$". Clearly, for any given natural number n, such a proposition $P(n)$ is either true or false. We want to show that $P(n)$ is true for all natural numbers n. This however follows on applying the Principle of Mathematical Induction, given that we have noted that $P(1)$ is true, and have demonstrated that if $P(m)$ is true for any natural number m then $P(m + 1)$ is also true.

1.3 Some examples of proofs using the Principle of Mathematical Induction

Example We claim that

$$\sum_{i=1}^{n} i = \tfrac{1}{2}n(n + 1)$$

for all natural numbers n, where

$$\sum_{i=1}^{n} i = 1 + 2 + \cdots + n.$$

We prove this result using the Principle of Mathematical Induction.

For any natural number n let $P(n)$ denote the proposition

$$\text{“}\sum_{i=1}^{n} i = \tfrac{1}{2}n(n + 1)\text{”}.$$

One can easily see that the proposition $P(1)$ is true, since both sides of the above identity reduce to the value 1 in this case.

Suppose that $P(m)$ is true for some natural number m. Then

$$\sum_{i=1}^{m} i = \tfrac{1}{2}m(m+1).$$

But then

$$\sum_{i=1}^{m+1} i = \sum_{i=1}^{m} i + (m+1) = \tfrac{1}{2}m(m+1) + (m+1) = \tfrac{1}{2}(m+1)(m+2),$$

and therefore the proposition $P(m+1)$ is also true. We can therefore conclude from the Principle of Mathematical Induction that $P(n)$ is true for all natural numbers, which is the result we set out to prove.

Example We prove by induction on n that

$$\sum_{i=1}^{n} i^2 = \tfrac{1}{6}n(n+1)(2n+1)$$

for all natural numbers n, where

$$\sum_{i=1}^{n} i^2 = 1^2 + 2^2 + \cdots + n^2.$$

To achieve this, we have to verify that the formula holds when $n = 1$, and that if the formula holds when $n = m$ for some natural number m, then the formula holds when $n = m + 1$.

The formula does indeed hold when $n = 1$, since $1 = \frac{1}{6} \times 1 \times 2 \times 3$.

Suppose that the formula holds when $n = m$. Then

$$\sum_{i=1}^{m} i^2 = \tfrac{1}{6}m(m+1)(2m+1).$$

But then

$$\begin{aligned}
\sum_{i=1}^{m+1} i^2 &= \sum_{i=1}^{m} i^2 + (m+1)^2 \\
&= \tfrac{1}{6}m(m+1)(2m+1) + (m+1)^2 \\
&= \tfrac{1}{6}(m+1)\left(m(2m+1) + 6(m+1)\right) = \tfrac{1}{6}(m+1)\left(2m^2 + 7m + 6\right) \\
&= \tfrac{1}{6}(m+1)(m+2)(2m+3),
\end{aligned}$$

and therefore the formula holds when $n = m+1$. The required result therefore follows using the Principle of Mathematical Induction.

Example We prove by induction on n that

$$1 \cdot 4 + 2 \cdot 5 + 3 \cdot 6 + \cdots + n(n+3) = \tfrac{1}{3}n(n+1)(n+5).$$

for all natural numbers n. The left hand side of the above identity may be written as $\displaystyle\sum_{i=1}^{n} i(i+3)$.

The required identity

$$\sum_{i=1}^{n} i(i+3) = \tfrac{1}{3}n(n+1)(n+5)$$

holds when $n = 1$, since both sides are then equal to 4. Suppose that this identity holds when n is equal to some natural number m, so that

$$\sum_{i=1}^{m} i(i+3) = \tfrac{1}{3}m(m+1)(m+5).$$

Then

$$\begin{aligned}
\sum_{i=1}^{m+1} i(i+3) &= \sum_{i=1}^{m} i(i+3) + (m+1)(m+4) \\
&= \tfrac{1}{3}m(m+1)(m+5) + (m+1)(m+4) \\
&= \tfrac{1}{3}(m+1)\Big(m(m+5) + 3(m+4)\Big) \\
&= \tfrac{1}{3}(m+1)(m^2+8m+12) \\
&= \tfrac{1}{3}(m+1)(m+2)(m+6),
\end{aligned}$$

and therefore the required identity $\displaystyle\sum_{i=1}^{n} i(i+3) = \tfrac{1}{3}n(n+1)(n+5)$ holds when $n = m+1$. It now follows from the Principle of Mathematical Induction that this identity holds for all natural numbers m.

Example We can use the Principle of Mathematical Induction to prove that

$$\sum_{k=1}^{n} 5^k k = \frac{5}{16}\Big((4n-1)5^n + 1\Big).$$

for all natural numbers n. This equality holds when $n = 1$, since both sides are then equal to 5. Suppose that the equality holds when $n = m$ for some natural number m, so that

$$\sum_{k=1}^{m} 5^k k = \frac{5}{16}\Big((4m-1)5^m + 1\Big).$$

Then

$$\begin{aligned}
\sum_{k=1}^{m+1} 5^k k &= \sum_{k=1}^{m} 5^k k + 5^{m+1}(m+1) \\
&= \frac{5}{16}\Big((4m-1)5^m + 1\Big) + 5^{m+1}(m+1) \\
&= \frac{5}{16}\Big((4m-1)5^m + 1 + 16(m+1)5^m\Big) \\
&= \frac{5}{16}\Big((20m+15)5^m + 1\Big) = \frac{5}{16}\Big((4m+3)5^{m+1} + 1\Big) \\
&= \frac{5}{16}\Big((4(m+1)-1)5^{m+1} + 1\Big).
\end{aligned}$$

and thus the equality holds when $n = m+1$. It follows from the Principle of Mathematical Induction that the equality holds for all natural numbers n.

Example We now use Principle of Mathematical Induction to prove that $6^n - 1$ is divisible by 5 for all natural numbers n. The result is clearly true when $n = 1$. Suppose that the result is true when $n = m$ for some natural number m. Then $6^m - 1$ is divisible by 5. But then

$$6^{m+1} - 1 = 6^{m+1} - 6^m + (6^m - 1) = 5 \times 6^m + (6^m - 1),$$

and therefore $6^{m+1} - 1$ is also divisible by 5. It therefore follows that $6^n - 1$ is divisible by 5 for all natural numbers n.

Example Given any two positive integers n and k we define the *binomial coefficient* $\binom{n}{k}$ to be the number of ways of choosing k distinct objects from a collection consisting of n objects. We also define $\binom{n}{0} = 1$ for all natural numbers n, and we define

$$\binom{n}{k} = 0 \text{ whenever } k < 0.$$

Note that $\binom{n}{n} = 1$ (since the entire collection can be selected in exactly one way), and that $\binom{n}{k} = 0$ when $k > n$ (since it is clearly impossible to select more than n distinct objects from a collection consisting of n objects).

We wish to prove that

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \text{ whenever } 0 \leq k \leq n$$

(where $0! = 1$ and where, for each natural number n, $n!$ (n factorial) denotes the product $1 \times 2 \times 3 \times \cdots \times n$ of all the natural numbers between 1 and n). We shall prove this result using the Principle of Mathematical Induction.

First, though, we derive a *recursion formula* for the binomial coefficients.

We are interested in the number $\binom{n}{k}$ of ways of choosing k objects from a collection consisting of n objects, in the case where $n > 1$. Let us suppose for the sake of argument that those n objects are coloured balls. Moreover let us suppose that exactly one of those balls is coloured black, and that the remaining balls are coloured red. There are then two distinct types of choices that we can make. We can make a choice consisting entirely of red balls: let us refer to such a choice as a *type I* choice. Alternatively we can make a choice consisting of the black ball together with $k-1$ red balls: let us refer to such a choice as a *type II* choice. A *type I* choice requires us to choose k red balls from a collection of $n-1$ red balls, and there are $\binom{n-1}{k}$ such choices. A *type II* choice requires us to choose $k-1$ red balls from a collection of $n-1$ red balls, and there are $\binom{n-1}{k-1}$ such choices. The total number of choices is obtained by adding together the number of *type I* choices and the number of *type II* choices. It follows that

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}.$$

(Note that the definitions we have made ensure that this formula also holds when $k = 0$, and indeed when $k < 0$.)

We now proceed to prove the required formula for the binomial coefficients, using the Principle of Mathematical Induction. Let $P(n)$ denote the proposition

$$\text{“}\binom{n}{k} = \frac{n!}{k!(n-k)!} \text{ whenever } 0 \leq k \leq n\text{”}$$

The proposition $P(1)$ asserts that $\binom{1}{0} = \binom{1}{1} = 1$, which is certainly true.

Now suppose that $P(n)$ is true for some natural number n. We show that $P(n+1)$ is true. If $P(n)$ is true and if the integer k satisfies $1 \leq k \leq n$ then

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \quad \text{and} \quad \binom{n}{k-1} = \frac{n!}{(k-1)!(n+1-k)!}.$$

It then follows from the recursion formula derived above that

$$\binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} = \frac{n!}{k!(n-k)!} + \frac{n!}{(k-1)!(n+1-k)!}.$$

But

$$\frac{1}{(n-k)!} = \frac{n+1-k}{(n+1-k)!} \quad \text{and} \quad \frac{1}{(k-1)!} = \frac{k}{k!}.$$

It follows that

$$\begin{aligned}\binom{n+1}{k} &= \frac{n!}{k!(n+1-k)!}\,((n+1-k)+k) \\ &= \frac{(n+1)!}{k!(n+1-k)!}\end{aligned}$$

The required identity $\binom{n+1}{k} = \frac{(n+1)!}{k!(n+1-k)!}$ holds also when $k = 0$ and $k = n+1$, since it is easily seen that both sides of the identity are equal to 1 in these cases. We conclude that if the proposition $P(n)$ is true for any natural number n then the proposition $P(n+1)$ is also true. We can therefore conclude from the Principle of Mathematical Induction that the proposition $P(n)$ is true for all natural numbers n, which is what we are required to prove.

Example We can use the Principle of Mathematical Induction to prove that $(2n)! < 4^n(n!)^2$ for all natural numbers n. This inequality holds when $n = 1$, since in that case $(2n)! = 2! = 2$ and $4^n(n!)^2 = 4$. Suppose that the inequality holds when $n = m$ for some natural number m. Then $(2m)! < 4^m(m!)^2$. Now

$$(2(m+1))! = (2m+2)! = (2m)!(2m+1)(2m+2).$$

Also

$$4^{m+1}((m+1)!)^2 = 4(4^m(m!)^2)(m+1)^2.$$

Moreover

$$(2m+1)(2m+2) < (2m+2)^2 = 4(m+1)^2.$$

On multiplying together the two inequalities

$$(2m)! < 4^m(m!)^2 \quad \text{and} \quad (2m+1)(2m+2) < 4(m+1)^2$$

(which we are allowed to do since the quantities on both sides of these inequalities are strictly positive), we find that

$$(2m)!(2m+1)(2m+2) < 4(4^m(m!)^2)(m+1)^2.$$

Thus if the inequality $(2n)! < 4^n(n!)^2$ holds when $n = m$ then it also holds when $n = m+1$. We conclude from the Principle of Mathematical Induction that it must hold for all natural numbers n.

Example We can use the Principle of Mathematical Induction to prove that

$$1^3 + 2^3 + 3^3 + \cdots + n^3 > \tfrac{1}{4}(n^4 + 2n^3)$$

for all natural numbers n. This inequality holds when $n = 1$, since the left hand side is then equal to 1, and the right hand side is equal to $\frac{3}{4}$. Suppose that the inequality holds when $n = m$ for some natural number m, so that

$$\sum_{i=1}^{m} i^3 > \tfrac{1}{4}(m^4 + 2m^3).$$

Then

$$\begin{aligned}
\sum_{i=1}^{m+1} i^3 &= \sum_{i=1}^{m} i^3 + (m+1)^3 \\
&> \tfrac{1}{4}(m^4 + 2m^3) + (m+1)^3 \\
&= \tfrac{1}{4}\Big(m^4 + 2m^3 + 4(m+1)^3\Big) \\
&= \tfrac{1}{4}\Big(m^4 + 6m^3 + 12m^2 + 12m + 4\Big)
\end{aligned}$$

Now

$$\begin{aligned}
(m+1)^4 + 2(m+1)^3 &= (m^4 + 4m^3 + 6m^2 + 4m + 1) \\
&\quad + (2m^3 + 6m^2 + 6m + 2) \\
&= m^4 + 6m^3 + 12m^2 + 10m + 3
\end{aligned}$$

But $12m + 4 > 10m + 3$ (since $m > 0$), and therefore

$$m^4 + 6m^3 + 12m^2 + 12m + 4 > (m+1)^4 + 2(m+1)^3.$$

It follows that

$$\sum_{i=1}^{m+1} i^3 > \tfrac{1}{4}\Big(m^4 + 6m^3 + 12m^2 + 12m + 4\Big) > \tfrac{1}{4}((m+1)^4 + 2(m+1)^3).$$

Thus if the inequality

$$\sum_{i=1}^{n} i^3 > \tfrac{1}{4}(n^4 + 2n^3)$$

holds when $n = m$ for some natural number m, then it also holds when $n = m+1$. It follows from the Principle of Mathematical Induction that this identity holds for all natural numbers n.

2 Sets and Functions

2.1 Sets

A *set* is a collection of entities. (This collection may be empty.) The entities belonging to a set are referred to as *elements* of the set. If a is an element of a set A then we denote this fact by writing $a \in A$.

Two sets are said to be identical, or to be equal to one another, if and only if they have the same elements. Thus if A and B denote sets, then $A = B$ if and only if every element of A is an element of B and every element of B is an element of A.

If we have a list of entities, we denote the set consisting of these entities by enclosing the list within braces $\{\ldots\}$. For example the set consisting of the colours red, green and blue can be denoted by $\{\text{red}, \text{green}, \text{blue}\}$.

Note that the order in which elements are specified in such a list is irrelevant. For example, the set consisting of the two people Alice and Bob may be written either as $\{\text{Alice}, \text{Bob}\}$ or as $\{\text{Bob}, \text{Alice}\}$. In other words,

$$\{\text{Alice}, \text{Bob}\} = \{\text{Bob}, \text{Alice}\}.$$

A set is said to be *finite* if it contains a finite number of elements. Otherwise the set is said to be *infinite*.

Example The set $\mathbb{N}$ consisting of all natural numbers is an infinite set, as is the set $\mathbb{Z}$ consisting of all integers.

One set, the *empty set*, deserves special mention. This set is denoted by $\emptyset$. It has no elements.

The elements of a given set may themselves be sets (and thus have elements of their own).

2.2 Unions, Intersections and Complements of Sets

Let A and B be sets. We define the *union* $A \cup B$ of A and B to be the set consisting of all elements that belong to A or to B (or to both). We define the *intersection* $A \cap B$ of A and B to be the set consisting of all elements that belong to both A and B. We also define $A \setminus B$ to be the set consisting of elements of A that do not belong to B. If every element of B belongs to A (so that B is a *subset* of A), then $A \setminus B$ is customarily referred to as the *complement* of B in A.

Example Let $A = \{1, 2, 3, 4, 5\}$ and $B = \{4, 5, 6, 7, 8\}$. Then

$$\begin{aligned} A \cup B &= \{1, 2, 3, 4, 5, 6, 7, 8\}, \\ A \cap B &= \{4, 5\}, \\ A \setminus B &= \{1, 2, 3\}, \\ B \setminus A &= \{6, 7, 8\}. \end{aligned}$$

Example Let $\mathbb{Z}$ be the set of all integers, and let $2\mathbb{Z}$ denote the set of all even integers (i.e., all integers that are divisible by two). Then $\mathbb{Z} \setminus 2\mathbb{Z}$ is the set of all odd integers (i.e., all integers that are not divisible by two). We see that $2\mathbb{Z} \cup (\mathbb{Z} \setminus 2\mathbb{Z}) = \mathbb{Z}$ (i.e., the set of integers is the union of the set of even integers and the set of even integers, or in other words, every integer is even or odd). Also $2\mathbb{Z} \cap (\mathbb{Z} \setminus 2\mathbb{Z}) = \emptyset$ (i.e., the intersection of the set of even integers and the set of odd integers is empty, or in other words, no integer is both even and odd).

One may also form unions and intersections of three or more sets. If A, B and C are sets, then $A \cup B \cup C$ denotes the union of the three sets A, B and C, and consists of all elements that belong either to A or to B or to C. Similarly $A \cap B \cap C$ denotes the intersection of the three sets A, B, C. An entity x is an element of the intersection $A \cap B \cap C$ if and only if it is an element of A and also of B and of C. Analogous notations are used for unions and intersections of four or more sets.

Let A, B and C be sets. One can readily verify the following identities:

$$\begin{aligned} A \cup A &= A, \\ A \cap A &= A, \\ A \cup B &= B \cup A, \\ A \cap B &= B \cap A, \\ (A \cup B) \cup C &= A \cup (B \cup C) = A \cup B \cup C, \\ (A \cap B) \cap C &= A \cap (B \cap C) = A \cap B \cap C, \\ A \cap (B \cup C) &= (A \cap B) \cup (A \cap C), \\ A \cup (B \cap C) &= (A \cup B) \cap (A \cup C), \\ (A \cap B) \cup (A \setminus B) &= A, \\ (A \cap B) \cap (A \setminus B) &= \emptyset, \\ A \cup B &= (A \cap B) \cup (A \setminus B) \cup (B \setminus A), \\ A \setminus (B \cup C) &= (A \setminus B) \cap (A \setminus C), \\ A \setminus (B \cap C) &= (A \setminus B) \cup (A \setminus C). \end{aligned}$$

Example Let us verify that $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ for all sets A, B and C. Now, given any sets D and E a standard and useful method for proving that they are in fact the same set is to show that every element of D belongs to E and that every element of E belongs to D. For then it follows that the sets D and E have the same elements, and therefore $D = E$.

So let A, B and C be sets, let $D = A \cap (B \cup C)$ and let $E = (A \cap B) \cup (A \cap C)$. Let x be an element of D. Then $x \in A$. Also either $x \in B$ or $x \in C$ (or both). If $x \in B$ then $x \in A \cap B$, (since we also know that $x \in A$). But every element of $A \cap B$ is an element of the union $(A \cap B) \cup (A \cap C)$, which is E. Therefore $x \in E$. Similarly if $x \in C$, then $x \in A \cap C$, and hence $x \in E$. Thus we have seen that an element of D belongs to E in each of the two cases when $x \in B$ and when $x \in C$. We conclude that every element of D belongs to E.

Now let x be an element of E. Then either $x \in A \cap B$ or $x \in A \cap C$. In the first case $x \in B$, and in the second case $x \in C$, so that in either case $x \in B \cup C$. Moreover $x \in A$ in both cases. It follows that every element of E belongs to the intersection of A and $B \cup C$. This intersection is the set D. Thus every element of E belongs to D.

We have shown that the sets D and E have the same elements. Therefore D and E are in fact the same set, and so $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

Example Let us verify that $A \setminus (B \cup C) = (A \setminus B) \cap (A \setminus C)$ for all sets A, B and C. Let x be an element of $A \setminus (B \cup C)$. We must show that x belongs to the set of the right hand side of the above equality. Now $x \in A \setminus (B \cup C)$, and therefore x belongs to A but does not belong to $B \cup C$. In particular, x does not belong to B, nor to C. It follows that $x \in A \setminus B$, and also $x \in A \setminus C$. But then $x \in (A \setminus B) \cap (A \setminus C)$. We have thus shown that every element of $A \setminus (B \cup C)$ is an element of $(A \setminus B) \cap (A \setminus C)$.

Now let x be any element of $(A \setminus B) \cap (A \setminus C)$. Then $x \in (A \setminus B)$ and $x \in (A \setminus C)$. The element therefore cannot belong to B. Nor can it belong to C. But $x \in A$. We conclude therefore that x is an element of A that does not belong to $B \cup C$. (Every element of $B \cup C$ must belong either to B or to C.) Thus any element x of $(A \setminus B) \cap (A \setminus C)$ belongs to $A \setminus (B \cup C)$. We conclude that the sets $A \setminus (B \cup C)$ and $(A \setminus B) \cap (A \setminus C)$ are in fact the same set, since we have shown that an element of either is an element of the other. Thus $A \setminus (B \cup C) = (A \setminus B) \cap (A \setminus C)$.

2.3 Subsets and Power Sets

Definition Let A and B be sets. We say that the set B is a *subset* of A if every element of B is an element of A. If B is a subset of A then we denote

this fact by writing either $B \subset A$ or $A \supset B$.

The empty set $\emptyset$ is a subset of every set. Morover any set is a subset of itself (i.e., $A \subset A$ for any set A). Thus a non-empty set A always has at least two subsets, namely $\emptyset$ and A itself.

Let A and B be sets. If $A \subset B$ and $B \subset A$ then $A = B$. For if $A \subset B$ and $B \subset A$ then every element of A is an element of B, and also every element of B is an element of A. But then the sets A and B have the same elements, and therefore these sets are in fact the same set.

Definition Let A be a set. The *power set* $\mathcal{P}A$ is the set whose elements are the subsets of A.

Example Let A be a set consisting of exactly one element a, so that $A = \{a\}$. Then the subsets of A are the empty set $\emptyset$ and A itself. It follows that the power set $\mathcal{P}A$ of A is given by $\mathcal{P}A = \{\emptyset, A\}$ in this case. Note that the set A has 1 element and that its power set $\mathcal{P}A$ has 2 elements.

Example Let $A = \{1, 2\}$. Then $\mathcal{P}A = \{\emptyset, \{1\}, \{2\}, \{1, 2\}\}$. Note that the set A has 2 elements and that its power set $\mathcal{P}A$ has 4 elements.

Example Let A be the set consisting of the three colours red, green and blue. Let us for convenience denote these colours by R, G and B. Thus $A = \{\mathrm{R}, \mathrm{G}, \mathrm{B}\}$. Going systematically through the subsets of A with 0, 1, 2, and 3 elements, we see that the power set of A is given by the following:

$$\mathcal{P}A = \{\emptyset, \{\mathrm{R}\}, \{\mathrm{G}\}, \{\mathrm{B}\}, \{\mathrm{G}, \mathrm{B}\}, \{\mathrm{B}, \mathrm{R}\}, \{\mathrm{R}, \mathrm{G}\}, \{\mathrm{R}, \mathrm{G}, \mathrm{B}\}\}\,.$$

Note that the set A has 3 elements and its power set $\mathcal{P}A$ has 8 elements.

Example Let A be a set consisting of the four elements a, b, c and d. Then the power set $\mathcal{P}A$ of A consists of the following subsets of A: the empty set $\emptyset$, $\{a\}$, $\{b\}$, $\{c\}$, $\{d\}$, $\{a, b\}$, $\{a, c\}$, $\{a, d\}$, $\{b, c\}$, $\{b, d\}$, $\{c, d\}$, $\{b, c, d\}$, $\{a, c, d\}$, $\{a, b, d\}$, $\{a, b, c\}$ and $\{a, b, c, d\}$. Thus the set A has one subset with no elements, four subsets with exactly one element, six subsets with exactly two elements, four subsets with exactly three elements, and one subset with exactly four elements. Note that the set A has 4 elements and its power set $\mathcal{P}A$ has 16 elements.

The pattern emerging from the above examples would lead one to conjecture the following theorem on the number of elements in the power set of a finite set, which we now proceed to state and prove.

Theorem 2.1 *If a finite set A has exactly n elements, then its power set $\mathcal{P}A$ has exactly 2^n elements.*

Proof Let A be a set with n elements, where $n > 0$. Choose an element a of A, and let B be the subset of A consisting of all elements of A apart from a (i.e., B is the complement $A \setminus \{a\}$ of $\{a\}$ in A). The set B has $n - 1$ elements. Now for each subset C of B there exist exactly two subsets of A whose intersection with B is the set C; these subsets are C itself and $C \cup \{a\}$ (i.e., the subset of A obtained by adjoining the element a to C). It follows that the set A has twice as many subsets as the set B.

If A has just one element then its power set $\mathcal{P}A$ has two elements. Indeed if $A = \{a\}$ then $\mathcal{P}A = \{\emptyset, A\}$.

An easy application of the Principle of Mathematical Induction proves that a if finite set has n elements then its power set has 2^n elements. Indeed this result holds for all sets with just one element, and if, for any natural number m, the result holds for all sets with m elements, then it also holds for all sets with $m+1$ elements, since we have already seen that the addition of an element to a set doubles the number of subsets which it contains. ∎

2.4 The Specification of Sets

We come now to consider a standard method for specifying sets in terms of the properties satisfied by their elements.

Suppose we wish to specify the subset of a given set A consisting of all elements of A that satisfy a given condition. Such a set is specified by the following:

$$\{a \in A : \textit{condition}\}$$

where '*condition*' is to be replaced in the above by the specific condition that an element a of the set A has to satisfy in order to belong to the subset being specified, as in the following examples.

Example Suppose we wish to specify the set consisting of all natural numbers greater than 7. This set can be specified as

$$\{n \in \mathbb{N} : n > 7\}.$$

Here $\mathbb{N}$ denotes the set of natural numbers. Note that this set can also be specified as

$$\{n \in \mathbb{Z} : n > 7\},$$

where $\mathbb{Z}$ denotes the set of integers (i.e., whole numbers). (Integers may be positive, negative or zero, but those integers n which also satisfy the condition $n > 7$ are positive, and are therefore natural numbers.)

Example The set of real numbers is denoted by $\mathbb{R}$. Therefore the set of real numbers whose squares are greater than 7 may be denoted by

$$\{x \in \mathbb{R} : x^2 > 7\}.$$

Example What is $\{x \in \mathbb{R} : x^2 < -7\}$?

Now the square of a real number x is always non-negative, whether x be positive, negative or zero. Therefore there are no real numbers x satisfying $x^2 < -7$. We conclude that $\{x \in \mathbb{R} : x^2 < -7\}$ is simply a somewhat complicated way of specifying the empty set $\emptyset$.

Example How many elements are there in the set $\{x \in \mathbb{R} : x^2 = 1\}$?

In other words, how many real numbers are there whose squares are equal to 1. There are exactly two, namely $+1$ and -1. Thus $\{x \in \mathbb{R} : x^2 = 1\} = \{-1, 1\}$. This set has two elements.

Example The set of real numbers that are less than -7 or greater than 4 may be denoted by

$$\{x \in \mathbb{R} : x < -7 \text{ or } x > 4\}$$

Example Note that $\{x \in \mathbb{R} : x < -7 \text{ and } x > 4\}$ is simply another somewhat complicated way of specifying the empty set.

Definition Let a and b be real numbers with $a \leq b$. We define

$$[a, b] = \{x \in \mathbb{R} : a \leq x \leq b\}, \quad (a, b) = \{x \in \mathbb{R} : a < x < b\},$$

$$[a, b) = \{x \in \mathbb{R} : a \leq x < b\}, \quad (a, b] = \{x \in \mathbb{R} : a < x \leq b\}.$$

Thus $[a, b]$ denotes the set consisting of all real numbers x that satisfy $a \leq x \leq b$, and the other sets are defined similarly. (Note that if $a = b$ then (a, b), $[a, b)$ and $(a, b]$ are all the empty set, and $[a, b]$ is the set $\{a\}$ consisting of the single element a.)

2.5 Binary Relations

A *binary* relation on a set specifies relations between pairs of elements from the set.

Example The relations $=$ ('equals'), $\neq$ ('not equal to'), $<$ ('less than'), $>$ ('greater than'), $\leq$ ('less than or equal to') and $\geq$ ('greater than or equal to') are all binary relations on the set $\mathbb{R}$ of real numbers.

Example Let A be a set, and let $\mathcal{P}A$ be the power set of A (i.e., the set whose elements are the subsets of A). Then $\subset$ is a binary relation on $\mathcal{P}A$, where two subsets B and C of A satisfy $B \subset C$ if and only if B is a subset of C.

If one has a relation R on a set A, then, given two elements x and y of A, either x is related to y, in which case we may write xRy, or else the element is not related to y.

Definition Let R be a relation on a set A.

The relation R is said to be *reflexive* when it has the following property: xRx for all elements x of the set A.

The relation R is said to be *symmetric* when it has the following property: if x and y are elements of the set A, and if xRy, then yRx.

The relation R is said to be *transitive* when it has the following property: if x, y and z are elements of the set A, and if xRy and yRz, then xRz.

An *equivalence relation* is a relation that is reflexive, symmetric and transitive.

Example The relation $<$ ('less than') on the set $\mathbb{R}$ of real numbers is neither reflexive nor symmetric, but it is transitive. Indeed there is no real number x satisfying $x < x$. Moreover there are no pairs of real numbers x and y satisfying both $x < y$ and $y < x$. However, if x, y and z are real numbers, and if $x < y$ and $y < z$, then $x < z$, and therefore the relation $<$ on $\mathbb{R}$ is transitive.

Example Let A be a non-empty set, and let $\mathcal{P}A$ be the power set of A. The relation $\subset$ on $\mathcal{P}A$ is reflexive and transitive, but is not symmetric. Indeed every subset of A is a subset of itself and therefore $B \subset B$ for all $B \in \mathcal{P}A$, showing that the relation $\subset$ on $\mathcal{P}A$ is reflexive. If B, C and D are subsets of A, and if $B \subset C$ and $C \subset D$, then $B \subset D$ (for if every element of B is an element of C and if every element of C is an element of D then clearly every element of B is an element of D), and therefore the relation $\subset$ on $\mathcal{P}A$ is transitive. It is not the case however that $B \subset C$ always implies that $C \subset B$. Indeed subsets B and C of A satisfy both $B \subset C$ and $C \subset B$ if and only if $B = C$. Thus the relation $\subset$ on $\mathcal{P}A$ is not symmetric.

Example The relation $=$ ('equals') on the set $\mathbb{R}$ of real numbers is an equivalence relation. However none of the relations $\neq$ ('not equal to'), $<$ ('less than'), $>$ ('greater than'), $\leq$ ('less than or equal to') or $\geq$ ('greater than or equal to') are equivalence relations on $\mathbb{R}$.

2.6 Congruences

Let m be a positive integer. We say that two integers x and y are *congruent modulo* m if $x - y$ is divisible by m. If x and y are congruent modulo m, then we denote this fact by writing

$$x \equiv y \pmod{m}.$$

Lemma 2.2 *Let m be a positive integer, and let x, y and z be integers. Then the following results hold:*

(i) $x \equiv x \pmod{m}$;

(ii) *if* $x \equiv y \pmod{m}$ *then* $y \equiv x \pmod{m}$;

(iii) *if* $x \equiv y \pmod{m}$ *and* $y \equiv z \pmod{m}$ *then* $x \equiv z \pmod{m}$.

The relation of congruence modulo m is thus reflexive, symmetric and transitive, and is therefore an equivalence relation on the set $\mathbb{Z}$ of integers.

Proof Clearly $x \equiv x \pmod{m}$ for any integer x, since $x - x = 0$, and 0 is divisible by any non-zero integer.

If $x \equiv y \pmod{m}$ then $x - y$ is divisible by m. But then $y - x$ is also divisible by m, and hence $y \equiv x \pmod{m}$.

If $x \equiv y \pmod{m}$ and $y \equiv z \pmod{m}$ then both $x - y$ and $y - z$ are divisible by m. But $x - z = (x - y) + (y - z)$ and the sum of two integers divisible by m is itself an integer divisible by m. Therefore $x - z$ is divisible by m, and hence $x \equiv z \pmod{m}$. ∎

Congruences play an important role in the study of the theory of numbers, and in applications of that theory to practical problems in areas such as cryptography.

One well known theorem, due to Pierre de Fermat, states that if p is any prime number then $x^p \equiv x \pmod{p}$ for all integers x. This result is sometimes referred to as *Fermat's Little Theorem*. This property of prime numbers is not shared by all natural numbers. For example $2^6 = 64$ and $64 \equiv 4 \pmod{6}$. But the numbers 2 and 4 are not congruent modulo 6 (since $4 - 2$ is not divisible by 6). Therefore the congruence $x^6 \equiv x \pmod{6}$ does not hold when $x = 2$.

2.7 Partitions and Equivalence Relations

Let A be a set. A *partition* of A is collection of subsets of A with the property that every element of A belongs to exactly one of the subsets in the collection.

Example Let $\mathbb{Z}$ be the set of integers, let O be the set of odd integers, and let E be the set of even integers. Every integer is either even or odd, and no integer is both even and odd. Therefore any integer belongs to exactly one of the sets O and E. Thus the collection consisting of the sets O and E is a partition of the set $\mathbb{Z}$ of integers.

There is a close connection between partitions and equivalence relations. We recall that an equivalence relation $\sim$ on a set A is a binary relation on A with the following properties:

(i) $x \sim x$ for all elements x of A (i.e., $\sim$ is *reflexive*);

(ii) if x and y are elements of A and if $x \sim y$ then $y \sim x$ (i.e., $\sim$ is *symmetric*);

(iii) if x, y and z are elements of A, and if $x \sim y$ and $y \sim z$ then $x \sim z$ (i.e., $\sim$ is *transitive*).

Definition Let $\sim$ be an equivalence relation on a set A, and let x be an element of A. The *equivalence class* $[x]$ of the element x is the subset of A defined as follows:

$$[x] = \{a \in A : a \sim x\}.$$

Example Let m be a positive integer. There is then an equivalence relation on the set $\mathbb{Z}$, where two elements x and y are related if and only if $x - y$ is divisible by m. (In other words, integers x and y are related if and only if $x \equiv y \pmod{m}$.) The equivalence class $[n]_m$ of an integer n thus consists of all integers x that are congruent to n modulo m. This equivalence class is referred to as the *congruence class* of n modulo m. An integer x belongs to the congruence class $[n]_m$ of n modulo m if and only if $x - n$ is divisible by m.

Now, given any integer x, exactly one of the integers

$$x, x-1, x-2, \ldots, x-m+1$$

between $x - m + 1$ and x is divisible by m. It follows that the integer x belongs to exactly one of the congruence classes $[0]_m, [1]_m, [2]_m, \ldots, [m-1]_m$. These congruence classes modulo m therefore constitute a partition of the set $\mathbb{Z}$ of integers.

Theorem 2.3 *Let $\sim$ be an equivalence relation on a set A. Then every element of A belongs to exactly one equivalence class. Thus the collection of equivalence classes is a partition of the set A.*

Proof Let x be an element of A. Then $x \sim x$ (since the relation $\sim$ is reflexive), and therefore $x \in [x]$. Thus every element x of A belongs to its own equivalence class $[x]$. We see from this that each element of A belongs to at least one equivalence class.

To complete the proof we must show that each element of A belongs to at most one equivalence class. Let x and y be elements of A. We shall show that if the equivalence classes $[x]$ and $[y]$ have at least one element in common then $[x] = [y]$.

Suppose then that there exists an element z of A that belongs to both $[x]$ and $[y]$. Then $z \sim x$ and $z \sim y$. But then $x \sim z$ (since the relation $\sim$ is symmetric), and hence $x \sim y$ (since $x \sim z$, $z \sim y$, and the relation $\sim$ is transitive). Moreover $y \sim x$, since the relation $\sim$ is symmetric. If a is an element of A and if $a \in [x]$ then $a \sim x$ and $x \sim y$, and therefore $a \in [y]$. Similarly if $a \in [y]$ then $a \sim y$ and $y \sim x$, and therefore $a \in [x]$. Thus every element of $[x]$ is an element of $[y]$, and every element of $[y]$ is an element of $[x]$. It follows that $[x] = [y]$.

We have proved that if equivalence classes $[x]$ and $[y]$ have at least one element in common then they coincide (i.e., they are in fact the same equivalence class). It follows that an element of A cannot belong to more than one equivalence class.

We have proved that every element of A belongs to exactly one equivalence class, since an element of A belongs to at least one equivalence class but cannot belong to more than one equivalence class. Thus the collection of equivalence classes is a partition of the set. ∎

Remark We have seen how every equivalence relation on a set gives rise to a partition of that set. On the other hand, any partition of the set gives rise to an equivalence relation on that set: two elements of the set are related if and only if they belong to the same subset in the partition. It follows that equivalence relations and partitions correspond to one another: to each equivalence relation on a set there is a corresponding partition of the set, and vice versa.

2.8 Partial Orders and Lattices

Definition Let A be a set. A binary relation R on A is said to be *anti-symmetric* if it has the following property:

if x and y are elements of A, and if xRy and yRx, then $x = y$.

Definition A *partial order* on a set is a relation on that set which is reflexive, transitive and anti-symmetric.

Let $\preceq$ denote a relation on a set A. We see that this relation is a partial order on the set A if and only if it has the following three properties:

(i) $x \preceq x$ for all elements x of A;

(ii) if x, y, and z are elements of A, and if $x \preceq y$ and $y \preceq z$, then $x \preceq z$;

(iii) if x and y are elements of A, and if $x \preceq y$ and $y \preceq x$, then $x = y$.

Example The relation $\leq$ ('less than or equal to') is a partial order on the set $\mathbb{R}$ of real numbers. (It clearly possesses all three properties listed above.) It is also a partial order when considered as a relation on the set $\mathbb{Z}$ of integers, or on the set $\mathbb{N}$ of natural numbers.

Example Let A be a set. The relation $\subset$ is a partial order on the power set $\mathcal{P}A$ of A, where subsets B and C satisfy $B \subset C$ if and only if B is a subset of C (i.e., if and only if every element of B belongs also to C).

Definition A *partially ordered set* (or *poset*) $(A, \preceq)$ consists of a set A, which is provided with a partial order $\preceq$ defined on the set.

Let $(A, \preceq)$ be a partially ordered set, and let B be a subset of A. An element l of A is said to be a *lower bound* of B if $l \preceq b$ for all elements b of B. An element l of A is said to be the *greatest lower bound* of B if l is a lower bound of B and if $l' \preceq l$ for all lower bounds l' of B. If such a greatest lower bound exists, we shall denote it by $\operatorname{glb} B$.

It is worth noting that a subset B of A can have at most one greatest lower bound. For if l and l' denote elements of A (not necessarily distinct), and if l and l' are greatest lower bounds of B then $l' \preceq l$ and $l \preceq l'$ and therefore $l = l'$ (since the relation $\preceq$ on A is anti-symmetric).

We can define in a similar fashion the notion of a *least upper bound* of a subset of A. An element u of A is said to be an *upper bound* of a subset B of A if $b \preceq u$ for all elements b of B. An element u of A is said to be the *least upper bound* of B if u is an upper bound of B and if $u \preceq u'$ for all upper bounds u' of B. A subset B of A can have at most one least upper bound. If such a least upper bound exists, we shall denote it by $\operatorname{lub} B$.

Example Consider the partially ordered set $(\mathbb{R}, \leq)$. Any finite subset B of $\mathbb{R}$ has both a greatest lower bound and a least upper bound. The greatest lower bound of B in this case is the smallest real number belonging to B, and the least upper bound is the largest real number belonging to B.

Example Let A be a set, and let $\mathcal{P}A$ be the power set of A (i.e., the set whose elements are the subsets of A). Then $(\mathcal{P}A, \subset)$ is a partially ordered set (i.e., the relation $\subset$ is a partial order on the power set $\mathcal{P}A$ of A). Given subsets B and C of A one can readily verify that

$$\text{glb}\{B, C\} = B \cap C, \quad \text{lub}\{B, C\} = B \cup C.$$

Indeed $B \cap C \subset B$ and $B \cap C \subset C$, and therefore $B \cap C$ is a lower bound of $\{B, C\}$. Moreover if D is any subset of A that is a lower bound of $\{B, C\}$ then $D \subset B$ and $D \subset C$, hence the elements of D must belong to both B and C, hence $D \subset B \cap C$. This shows that $\text{glb}\{B, C\} = B \cap C$. A similar argument shows that $\text{lub}\{B, C\} = B \cup C$.

Example Let $(\mathbb{N}, \leq)$ be the partially ordered set (poset) consisting of the set $\mathbb{N}$ of natural numbers, together with the usual partial order $\leq$. Let B be the subset of $\mathbb{N}$ consisting of all the even natural numbers (i.e., $B = \{2, 4, 6, 8, \ldots\}$). The set B has a greatest lower bound. Indeed $\text{glb}\, B = 2$. But the set B has no least upper bound. Indeed the set has no upper bound: no natural number has the property that it is greater than or equal to all even natural numbers.

Definition A partially ordered set $(A, \preceq)$ is said to be a *lattice* if, given any two elements x and y of the set A, there exists an element $\text{glb}\{x, y\}$ of A that is the greatest lower bound of the set $\{x, y\}$ and an element $\text{lub}\{x, y\}$ that is the least upper bound of the set $\{x, y\}$.

Example $(\mathbb{R}, \leq)$, $(\mathbb{Z}, \leq)$, $(\mathbb{N}, \leq)$ are lattices (where two numbers x and y satisfy $x \leq y$ if and only if x is less than or equal to y).

Example Let A be a set, and let $\mathcal{P}A$ denote the power set of A. Then $(\mathcal{P}A, \subset)$ is a lattice. Indeed, if B and C are elements of $\mathcal{P}A$ then they are subsets of A. Moreover we have already seen that

$$\text{glb}\{B, C\} = B \cap C, \quad \text{lub}\{B, C\} = B \cup C,$$

and $B \cap C$ and $B \cup C$ are elements of $\mathcal{P}$ (since they are obviously subsets of A). It follows that contains the greatest lower bound and least upper bound of the set $\{B, C\}$ for all elements B and C of $\mathcal{P}A$ (i.e., for all subsets B and C of A).

2.9 Cartesian Products of Sets

Let A and B be sets. The *Cartesian product* $A \times B$ of the sets A and B is defined to be the set of all *ordered pairs* (a, b) with $a \in A$ and $b \in B$.

Such an ordered pair (a, b) is comprised of two elements a and b, where the first element a is taken from the set A, and the second element b is taken from the set B. If (a_1, b_1) and (a_2, b_2) are ordered pairs of this type then $(a_1, b_1) = (a_2, b_2)$ if and only if $a_1 = a_2$ and $b_1 = b_2$.

Example Points of the plane are specified in Cartesian coordinates by means of ordered pairs (x, y), where x and y are real numbers. The set of such ordered pairs is the set $\mathbb{R} \times \mathbb{R}$ (the Cartesian product of two copies of the set $\mathbb{R}$ of real numbers).

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

$$A \times B = \{(1, 1), (2, 1), (3, 1), (1, 2), (2, 2), (3, 2)\}.$$

Note that, in this example, the number of elements of the set $A \times B$ (i.e., 6) is the product of the number of elements of A (i.e., 3) and the number of elements of B (i.e., 2).

Suppose that A and B are finite sets. Let m and n be the number of elements in A and B respectively. Then the number of elements of the Cartesian product $A \times B$ is mn. Indeed an element of $A \times B$ is an ordered pair (a, b) with $a \in A$ and $b \in B$. There are m ways to choose the element a from A, and, for each such choice, there are n ways to choose the element b from B.

One may form the Cartesian product of any number of sets. Suppose that $A_1, A_2, \ldots A_n$ are sets. The *Cartesian product* of these sets is the set $A_1 \times A_2 \times \cdots \times A_n$ consisting of all ordered n-tuples $(a_1, a_2, \ldots, a_n)$ with $a_i \in A_i$ for $i = 1, 2, \ldots, n$.

Example Points of three dimensional space are specified in Cartesian coordinates by means of ordered triples (x, y, z), where x, y and z are real numbers. The set of such ordered triples is the set $\mathbb{R} \times \mathbb{R} \times \mathbb{R}$.

Let $A_1, A_2, \ldots, A_n$ be sets, and let $(c_1, c_2, \ldots, c_n)$ and $(d_1, d_2, \ldots, d_n)$ be elements of the Cartesian product $A_1 \times A_2 \times \cdots \times A_n$ of these sets. Then $(c_1, c_2, \ldots, c_n) = (d_1, d_2, \ldots, d_n)$ if and only if $c_i = d_i$ for $i = 1, 2, \ldots, n$ (i.e., if and only if $c_1 = d_1$, $c_2 = d_2$, etc.).

A Cartesian product $A_1 \times A_2 \times \cdots \times A_n$ of finite sets $A_1, A_2, \ldots, A_n$ is itself a finite set: the number of elements of the Cartesian product is equal the product of the number of elements of the individual sets $A_1, A_2, \ldots, A_n$.

Example If the sets A, B and C have 3, 5 and 7 elements respectively then their Cartesian product has 105 elements, since $105 = 3 \times 5 \times 7$.

Example Suppose that one to construct a database containing information on students taking a course such as 2BA1. Each record in the database is to specify the student number, the name, and the degree programme being followed by the student. Let I be the set consisting of all strings of eight decimal digits, let N be a set containing all the student names, and let D be the set of all degree programmes taught at Trinity College Dublin. Then a record in the database determines an element of the set $I \times N \times D$, such as

$$(63009987, \text{Síle Ní Shé}, \text{CSLL German}).$$

The collection of all such records contained in the database can be viewed as a subset of the Cartesian product $I \times N \times D$ of the set I, N and D. The language of sets and Cartesian products is used in discussions of *relational databases*.

A subset of the Cartesian product $A_1 \times A_2 \times \cdots \times A_n$ of sets $A_1, A_2, \ldots, A_n$ is sometimes referred to as an *n-ary relation* on the sets $A_1, A_2, \ldots, A_n$.

2.10 Functions between Sets

Definition Let A and B be sets. A *function* $f: A \to B$ from A to B assigns to each element a of A an element $f(a)$ of B. The set A on which the function is defined is referred to as the *domain* of the function $f: A \to B$. The set B into which the domain is mapped by f is referred to as the *codomain* of the function f.

Example Let $\mathbb{R}$ be the set of real numbers. The function $q: \mathbb{R} \to \mathbb{R}$ defined by $q(x) = x^2$ for all real numbers x is a function from the set $\mathbb{R}$ of real numbers to itself.

Example There is a function $r: \mathbb{R} \setminus \{0\} \to \mathbb{R}$, where $r(x) = 1/x$ for all non-zero real numbers x. The domain of this function is the set $\mathbb{R} \setminus \{0\}$ of all non-zero real numbers (i.e., the set $\{x \in \mathbb{R} : x \neq 0\}$). The domain of this function cannot be extended to the entire set $\mathbb{R}$ of real numbers since the reciprocal of zero is not defined. According the above definition the value of a function must be defined at all elements of its domain.

Example Let A be the set of letters in the English alphabet (including both upper-case and lower-case letters). Then there is a function $f: A \to \mathbb{N}$ which sends each letter to its ASCII code. Then, for example, $f(\mathbf{A}) = 65$, $f(\mathbf{B}) = 66$, $f(\mathbf{a}) = 97$ and $f(\mathbf{b}) = 98$.

Given any set A, there is a function $1_A: A \to A$ from the set A to itself which sends each element a of A to itself. This function is referred to as the *identity function* on A.

Definition Let A and B be sets, and let $f: A \to B$ be a function from A to B. The *range* of the function f is the subset $f(A)$ of B defined by

$$f(A) = \{b \in B : b = f(a) \text{ for some } a \in A\}.$$

In other words, the *range* of a function is the set consisting of all elements of the codomain of the function that are images under the function of elements of its domain.

Definition Let A be a set. A *Boolean function* on A is a function $f: A \to \{T, F\}$ whose domain is A and whose codomain is the set $\{T, F\}$ whose elements are the *truth values* $T = \text{true}$ and $F = \text{false}$.

2.11 Compositions of Functions

Let A, B and C be sets, let $f: A \to B$ be a function A to B, let $g: B \to C$ be a function from B to C. Then there is a function $g \circ f: A \to C$ obtained by composing the functions f and g. This function is defined at each element a of A by the formula $(g \circ f)(a) = g(f(a))$. (In other words, in order to apply the composition function $g \circ f$ to an element a of A, we first apply the function f to the element a, and then we apply the function g to the resulting element $f(a)$ of B to obtain an element $g(f(a))$ of C.

Example Let $\mathbb{R}$ denote the set of real numbers, and let $f: \mathbb{R} \to \mathbb{R}$ and $g: \mathbb{R} \to \mathbb{R}$ be the functions defined by $f(x) = (x + 1)^2$ and $g(x) = \sin x$ for all real numbers x. Then $g \circ f = h$ where $h: \mathbb{R} \to \mathbb{R}$ is the function defined by $h(x) = \sin(x + 1)^2$ for all real numbers x. Also $f \circ g = k$, where $k: \mathbb{R} \to \mathbb{R}$ is the function defined by $k(x) = (\sin x + 1)^2$ for all real numbers x.

Remark Note that '$g \circ f$' denotes the composition function 'f followed by g'. The functions are specified in this order (which may at first seem odd) in order that $(g \circ f)(a) = g(f(a))$ for all elements a of the domain A of the function f.

2.12 The Graph of a Function

Let A and B be sets. To every function $f: A \to B$ from A to B there corresponds a subset $\Gamma(f)$ of the Cartesian product $A \times B$, where

$$\Gamma(f) = \{(a, b) \in A \times B : b = f(a)\}.$$

Mathematicians often refer to the subset of $A \times B$ corresponding to a function $f: A \to B$ as the *graph* of the function. The following example suggests the reason for this terminology.

Example Let $q: \mathbb{R} \to \mathbb{R}$ be the function from the set $\mathbb{R}$ of real numbers to itself defined such that $q(x) = x^2$ for all real numbers x. The graph of this function is the subset of $\mathbb{R} \times \mathbb{R}$ given by

$$\{(x, y) \in \mathbb{R} \times \mathbb{R} : y = x^2\}.$$

Note that this subset consists of the Cartesian coordinates of the points of the plane that lie on the curve that represents the graph of the given function.

Whilst every function from A to B determines a corresponding subset $\Gamma(f)$ of $A \times B$, it is not possible to obtain every subset of $A \times B$ in this fashion. Indeed it is easy to see that a subset R of $A \times B$ is the graph of some function $f: A \to B$ if and only if, for every element a of A, there exists exactly one element b of B for which $(a, b) \in R$. If the subset R of $A \times B$ has this property, then the corresponding function $f: A \to B$ is characterized by the property that, for each element a of A, $f(a)$ is the unique element of B for which $(a, f(a)) \in R$.

Remark In some books, including many textbooks on discrete mathematics written for students of computer science, a function from a set A to a set B is formally defined as a subset of the Cartesian product $A \times B$ with the property that for each element a of A there exists exactly one element b of B for which the ordered pair (a, b) belongs to the given subset. In essence, in this approach, functions are being identified with their graphs.

2.13 The Inverse of a Function

Definition Let A and B be sets, and let $f: A \to B$ be a function from A to B. A function $g: B \to A$ from B to A is said to be the *inverse* of the function f if $g(f(a)) = a$ for all elements a of A and $f(g(b)) = b$ for all elements b of B. If there exists a function $g: B \to A$ that is the inverse of $f: A \to B$, then the function f is said to be *invertible* and the inverse of a function $f: A \to B$ is denoted by $f^{-1}: B \to A$.

Example Let $\mathbb{R}^+$ denote the set of all non-negative real numbers, and let $q: \mathbb{R}^+ \to \mathbb{R}^+$ denote the function defined by $q(x) = x^2$ for each non-negative real number x. This function is invertible, and its inverse $q^{-1}: \mathbb{R}^+ \to \mathbb{R}^+$ is given by $q^{-1}(x) = \sqrt{x}$, where, for each non-negative real number x, $\sqrt{x}$ denotes the unique non-negative real number that is a square root of x.

Example Let A be the set of letters in the English alphabet (including both upper-case and lower-case letters), and let

$$I = \{n \in \mathbb{N} : 65 \leq n \leq 90 \text{ or } 97 \leq n \leq 122\}.$$

There is then a function $f: A \to I$ that sends each letter of the alphabet to its ASCII code. The inverse function $f^{-1}: I \to A$ sends each natural number within the specified ranges to the letter of the English alphabet which it represents. Thus, for example, $f^{-1}(65) = \mathbf{A}$, $f^{-1}(66) = \mathbf{B}$, $f^{-1}(90) = \mathbf{Z}$, $f^{-1}(97) = \mathbf{a}$, $f^{-1}(98) = \mathbf{b}$ and $f^{-1}(122) = \mathbf{z}$.

2.14 Injective, Surjective and Bijective Functions

Many functions are not invertible. The following example illustrates some of the reasons why certain functions may not be invertible.

Example Let W be the set of all English words occurring as headwords in some specified dictionary, let $\mathbb{N}$ denote the set of natural numbers and let $\lambda: W \to \mathbb{N}$ denote the function that sends each word to its length. (Thus, for example, $\lambda(\mathbf{to}) = 2$ and $\lambda(\mathbf{indecipherable}) = 14$.) This function $\lambda: W \to \mathbb{N}$ is not invertible.

One feature of this function which results in its not being invertible is the fact that there are natural numbers that are the image of more than one word. For example

$$\lambda(\mathbf{to}) = \lambda(\mathbf{by}) = \lambda(\mathbf{at}) = 2.$$

$$\lambda(\mathbf{physical}) = \lambda(\mathbf{computer}) = 8.$$

If one were to seek to define function $\mu: \mathbb{N} \to W$ that was the inverse of $\lambda: W \to \mathbb{N}$ then one would run into problems in seeking to define values such as $\mu(2)$ and $\mu(8)$. Indeed if such an inverse function $\mu: \mathbb{N} \to W$ were to exist, then it would have to satisfy $\mu(\lambda(\alpha)) = \alpha$ for all words α in the dictionary. In particular we would have $\mu(\lambda(\mathbf{physical})) = \mathbf{physical}$ and $\mu(\lambda(\mathbf{computer})) = \mathbf{computer}$. But $\mu(\lambda(\mathbf{physical})) = \mu(8)$, and $\mu(\lambda(\mathbf{computer})) = \mu(8)$, and therefore the inverse function $\mu: \mathbb{N} \to W$ would also have to satisfy $\mu(\lambda(\mathbf{physical})) = \mu(\lambda(\mathbf{computer}))$, and therefore the words '**physical**' and '**computer**' would have to be identical, which is clearly not the case. This demonstrates the impossibility of finding an inverse function to λ.

Another type of problem can also arise in seeking to define an inverse $\mu: \mathbb{N} \to W$ to the function $\lambda: W \to \mathbb{N}$. How do we define $\mu(1000)$? Now the inverse function μ would have to satisfy $\lambda(\mu(n)) = n$ for all natural numbers, and in particular would have to satisfy $\lambda(\mu(1000)) = 1000$. Therefore $\mu(1000)$

would have to be a headword in the specified dictionary with 1000 letters! We take it for granted that no such headword exists.

Definition Let A and B be sets, and let $f: A \to B$ be a function from A to B. We say that the function f is *injective* if $f(x) \neq f(y)$ for all elements x and y of A with $x \neq y$. We say that the function f is *surjective* if, given any element b of B, there exists some element a of A such that $f(a) = b$. We say that the function f is *bijective* if it both injective and surjective.

Thus a function is injective if and only if distinct elements of its domain get mapped to distinct elements of its codomain. A function is surjective if every element of the codomain is the image of some element of the domain.

Example Let $\mathbb{R}^+$ denote the set of non-negative real numbers, and let $q: \mathbb{R}^+ \to \mathbb{R}^+$ be the function given by $q(x) = x^2$ for all non-negative real numbers x. Let x and y be non-negative real numbers. If $x < y$ then $x^2 < y^2$. If $x > y$ then $x^2 > y^2$. But if $x \neq y$ then either $x < y$ or $x > y$. It follows that if $x \neq y$ then $x^2 \neq y^2$. The function $q: \mathbb{R}^+ \to \mathbb{R}^+$ is therefore injective. Also, given any non-negative real number x, there exists a non-negative real number $\sqrt{x}$ whose square is equal to x. It follows that the function $q: \mathbb{R}^+ \to \mathbb{R}^+$ is both injective and surjective. It is therefore bijective. This function also has an inverse $q^{-1}: \mathbb{R}^+ \to \mathbb{R}^+$, where $q^{-1}(x) = \sqrt{x}$ for all non-negative real numbers x.

Example Let $s: \mathbb{R} \to \mathbb{R}$ by the function given by $s(x) = x^2$ for all real numbers x. This function is not injective. For example, -2 and 2 are distinct elements of $\mathbb{R}$, but $s(-2) = 4 = s(2)$. Moreover the function is not surjective, since any negative real number such as -4 is not in the range of the function. This function $s: \mathbb{R} \to \mathbb{R}$ is neither injective nor surjective. Moreover one can easily satisfy oneself that it does not have an inverse. (Such an inverse, were it to exist, would have to be defined for *all* real numbers, not merely the non-negative ones.)

Remark Note that the expressions defining the values $q(x)$ and $s(x)$ of the functions of the previous two examples are the same, but these two functions have different domains and different codomains, and are therefore regarded as being different functions. In determining whether or not functions are injective or surjective, it is crucial to take into account the domain and codomain given in the specification of the function.

One can readily verify that the composition of two injections is itself an injection, and that the composition of two surjections is itself a surjection. It follows directly that the composition of two bijections is a bijection.

Theorem 2.4 *A function $f: A \to B$ is invertible if and only if it is both injective and surjective.*

Proof First we show that an invertible function must be both injective and surjective. Suppose that the function $f: A \to B$ has an inverse $g: B \to A$. Then $g(f(a)) = a$ for all elements a of the domain A, and $f(g(b)) = b$ for all elements b of the codomain B. Let x and y be elements of A. If $f(x) = f(y)$ then $x = g(f(x)) = g(f(y)) = y$. Thus $f(x)$ and $f(y)$ cannot be equal unless $x = y$. It follows that if $x \neq y$ then $f(x) \neq f(y)$. We see therefore that an invertible function must be injective.

An invertible function must also be surjective. For if $g: B \to A$ is an inverse of $f: A \to B$ then $f(g(b)) = b$ for all elements b of the codomain B, and thus there exists at least one element of the domain, namely $g(b)$, which is mapped by f to the element b.

We have now shown that an invertible function must be both injective and surjective. It remains to show that a function that is both injective and surjective is invertible.

Let $f: A \to B$ be a function that is both injective and surjective. Let b be an element of the set B. There exists at least one element x of A satisfying $f(x) = b$, since the function f is surjective. If y is an element of A and if $y \neq x$, then $f(y) \neq f(x)$, because the function f is injective, and therefore $f(y) \neq b$. We conclude that, for each element b of B, there exists exactly one element x of the set A satisfying $f(x) = b$; let us denote this element by $g(b)$. We obtain in this way a function $g: B \to A$ such that, for each element b of B, $g(b)$ is the unique element x of A satisfying $f(x) = b$.

Clearly $f(g(b)) = b$ for all elements b of B. In order to prove that the function $g: B \to A$ is the inverse of $f: A \to B$, we must also prove that $g(f(a)) = a$ for all elements a of A. Let a be an element of the set A. Now $f(g(b)) = b$ for all elements b of B; letting $b = f(a)$, we see that $f(g(f(a))) = f(a)$. But then $g(f(a))$ and a are both elements of A that are mapped by f to the element $f(a)$ of B. It follows that $g(f(a)) = a$, since the function f is injective. We have thus shown that $g(f(a)) = a$ for any element a of the domain A of the function f. We conclude that the function $g: B \to A$ is indeed the inverse of $f: A \to B$, and thus the function f is invertible, as required. ∎

The above theorem shows that a function between sets is invertible if and only if it is a bijection.

Example Let $q: [-3, 1] \to [0, 9]$ be the function defined by $q(x) = x^2$ for all $x \in [-3, 1]$, where

$$[-3, 1] = \{x \in \mathbb{R} : -3 \leq x \leq 1\} \quad \text{and} \quad [0, 9] = \{x \in \mathbb{R} : 0 \leq x \leq 9\}.$$

(We recall that, given any real numbers a and b satisfying $a \leq b$, the set of real numbers x satisfying $a \leq x \leq b$ is denoted by $[a, b]$.) The function $q\colon [-3, 1] \to [0, 9]$ is surjective, since for each real number y satisfying $0 \leq y \leq 9$, there exists at least one real number x satisfying $-3 \leq x \leq 1$ such that $q(x) = y$; one such real number x is given by $x = -\sqrt{y}$, where $\sqrt{y}$ denotes the positive square root of y. However the function q is not injective. Indeed $q(1) = q(-1) = 1$. The function $q\colon [-3, 1] \to [0, 9]$ is therefore not bijective, and hence is not invertible.

Example Let $f\colon [0, 2] \to [0, 2]$ and $g\colon [0, 2] \to [0, 2]$ be the functions defined by

$$f(x) = \begin{cases} x^2 & \text{if } 0 \leq x \leq 1; \\ 3 - x & \text{if } 1 < x \leq 2; \end{cases}$$

$$g(x) = \begin{cases} x^2 & \text{if } 0 \leq x < 1; \\ 3 - x & \text{if } 1 \leq x \leq 2. \end{cases}$$

The function $f\colon [0, 2] \to [0, 2]$ is not injective since $f(1) = f(2) = 1$. This function is not surjective, since there is no element x of the domain $[0, 2]$ for which $f(x) = 2$. The function f is thus not bijective, and hence is not invertible. The function $g\colon [0, 2] \to [0, 2]$, on the other hand, is invertible, with inverse given by

$$g^{-1}(x) = \begin{cases} \sqrt{x} & \text{if } 0 \leq x < 1; \\ 3 - x & \text{if } 1 \leq x \leq 2. \end{cases}$$

It follows from this that the function $g\colon [0, 2] \to [0, 2]$ must be both injective and surjective.

2.15 Partial Mappings

There is a generalization of the concept of a function between sets that is used in theoretical computer science. This is the concept of a *partial mapping*.

A *partial mapping* (or *partial function*) $f\colon A \nrightarrow B$ associates to some (but not necessarily all) elements a of A a corresponding element $f(a)$ of B. Partial mappings may be used in computer science to represent functions that are not defined for all their input values. For example, suppose that one has an algorithm which takes as input a natural number n and which, if it terminates, returns some other natural number $f(n)$. However there may be values of n for which the algorithm does not terminate, and for such values, the return value $f(n)$ is not defined. This situation is then represented by a partial mapping $f\colon \mathbb{N} \nrightarrow \mathbb{N}$.

The *domain* of a partial mapping $f\colon A \nrightarrow B$ is defined to be the subset of A consisting of all elements a of A for which $f(a)$ is defined. The *range* of the partial mapping $f\colon A \nrightarrow B$ is defined to be the subset of B consisting of all elements of B that are of the form $f(a)$ for some element a of the domain of the partial mapping.

A partial mapping $f\colon A \nrightarrow B$ is said to be *total* if its domain is the whole of A. Thus a partial mapping $f\colon A \nrightarrow B$ is total if and only if it is in fact a function from A to B.

A theory of partial mappings may be developed that generalizes the theory of functions between sets.

3 Graph Theory

3.1 Undirected Graphs

An *undirected graph* can be thought of as consisting of a finite set V of points, referred to as the *vertices* of the graph, together with a finite set E of *edges*, where each edge joins two distinct vertices of the graph.

We now proceed to formulate the definition of an undirected graph in somewhat more formal language.

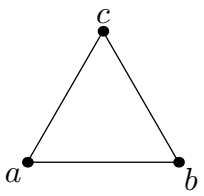
Let V be a set. We denote by V_2 the set consisting of all subsets of V with exactly two elements. Thus, for any set V,

$$V_2 = \{A \in \mathcal{P}V : |A| = 2\},$$

where $\mathcal{P}V$ denotes the power set of V (i.e., the set consisting of all subsets of V), and $|A|$ denotes the number of elements in a subset A of V.

Definition An *undirected graph* (V, E) consists of a finite set V together with a subset E of V_2 (where V_2 is the set consisting of all subsets of V with exactly two elements). The elements of V are the *vertices* of the graph; the elements of E are the *edges* of the graph.

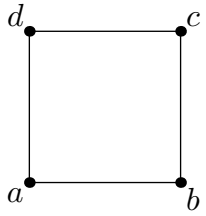
Example Let a, b and c label the three vertices of a triangle in the plane. Then there is an undirected graph (V, E) which consists of the vertices and edges of this triangle.



Here

$$\begin{aligned} V &= \{a, b, c\}; \\ E &= \{\{a, b\}, \{b, c\}, \{c, a\}\}. \end{aligned}$$

Example Let a, b, c and d label the four vertices of a square in the plane (labelled in cyclic order around the square). Then there is an undirected graph (V, E) which consists of the vertices and edges of this square.



Here

$$\begin{aligned} V &= \{a, b, c, d\}; \\ E &= \{\{a, b\},\ \{b, c\},\ \{c, d\},\ \{d, a\}\}. \end{aligned}$$

Note that, in this example, not every subset of V_2 with exactly two elements is an edge of the graph. (Indeed the diagonals $\{a, c\}$ and $\{b, d\}$ are not edges of this graph.)

Let (V, E) be an undirected graph. In order to simplify notation, we shall often denote by $a\,b$ an edge $\{a, b\}$ of the graph whose endpoints are the vertices a and b.

Definition A graph is said to be *trivial* if it consists of a single vertex.

We may denote a graph by a single letter such as G. Writing $G = (V, E)$ indicates that V is the set of vertices and E is the set of edges of some graph G.

3.2 Incidence and Adjacency

Definition If v is a vertex of some graph, if e is an edge of the graph, and if $e = v\,v'$ for some vertex v' of the graph, then the vertex v is said to be *incident* to the edge e, and the edge e is said to be *incident* to the vertex v.

(We see therefore that an edge of a graph is *incident* to a vertex of the graph, and the vertex is *incident* to the edge, if and only if the vertex is one of the endpoints of the edge.)

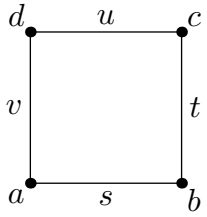
Definition Two distinct vertices v and v' of a graph (V, E) are said to be *adjacent* if and only if $v\,v' \in E$.

(We see therefore that two distinct vertices of a graph are *adjacent* if and only if they are the endpoints of an edge of the graph.)

3.3 Incidence and Adjacency Tables and Matrices

The following example illustrates how we may associate incidence and adjacency tables or matrices with graphs.

Example Let a, b, c and d represent the four vertices of a square in the plane, and consider the graph consisting of the vertices and edges of this square. Let s, t, u and v denote the four edges of the square, where $s = a\,b$, $t = b\,c$, $u = c\,d$ and $v = d\,a$.



The incidence relations between the vertices a, b, c and d and the edges s, t, u and v can be expressed by the following table:

	s	t	u	v
a	1	0	0	1
b	1	1	0	0
c	0	1	1	0
d	0	0	1	1

Such a table is known as the *incidence table* for the graph.

If a vertex is incident to an edge then the corresponding entry in the table has the value 1; otherwise that entry has the value 0.

If the vertices are ordered (as first vertex, second vertex, etc.) and if the edges are also ordered, then this information may be encoded in a matrix, known as an *incidence matrix*. In this example, if the vertices are ordered as a, b, c, d (so that a is the first vertex, b is the second vertex, c is the third vertex, and d is the fourth vertex), and if the edges are ordered as s, t, u, v, then the corresponding incidence matrix is

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

Definition Let (V, E) be a graph with m vertices and n edges. Let the vertices be ordered as $v_1, v_2, \ldots, v_m$, and let the edges be ordered as $e_1, e_2, \ldots, e_n$. The *incidence matrix* for such a graph then takes the form

$$\begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{pmatrix},$$

where the entry a_{ij} in the ith row and jth column has the value 1 if the ith vertex is incident to the jth edge, but has the value 0 otherwise.

One may introduce in a similar fashion the *adjacency table* and the *adjacency matrix* of a graph.

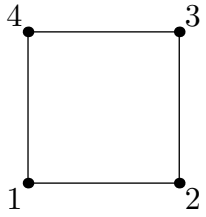
Definition Let (V, E) be a graph with m vertices, and let the vertices be ordered as $v_1, v_2, \ldots, v_m$. The *adjacency matrix* for such a graph then takes the form

$$\begin{pmatrix} b_{11} & b_{12} & \ldots & b_{1m} \\ b_{21} & b_{22} & \ldots & b_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & a_{m2} & \ldots & b_{mm} \end{pmatrix},$$

where the entry b_{ij} in the ith row and jth column has the value 1 if the ith vertex is adjacent to the jth vertex but has the value 0 otherwise.

Note that the adjacency matrix of any (undirected) graph is symmetric: $b_{ij} = b_{ji}$ for all indices i and j, where b_{ij} denotes the entry in the ith row and jth column of the adjacency matrix.

Example Consider the graph consisting of the vertices and edges of a square in the plane. Suppose that the vertices are ordered in anticlockwise order around the square, starting from some chosen vertex of the square.



Then the adjacency matrix for this graph is the matrix

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

3.4 Complete Graphs

Definition A graph (V, E) is said to be *complete* if and only if, $\{v, v'\} \in E$ for all $v \in V$ and $v' \in V$ satisfying $v \neq v'$.

(Thus a graph is *complete* if and only if any two distinct vertices of the graph are the endpoints of an edge of the graph.)

A complete graph with n vertices is denoted by K_n.

3.5 Bipartite Graphs

Definition A graph (V, E) is said to be *bipartite* if there exist subsets V_1 and V_2, such that

(i) $V_1 \cup V_2 = V$;

(ii) $V_1 \cap V_2 = \emptyset$;

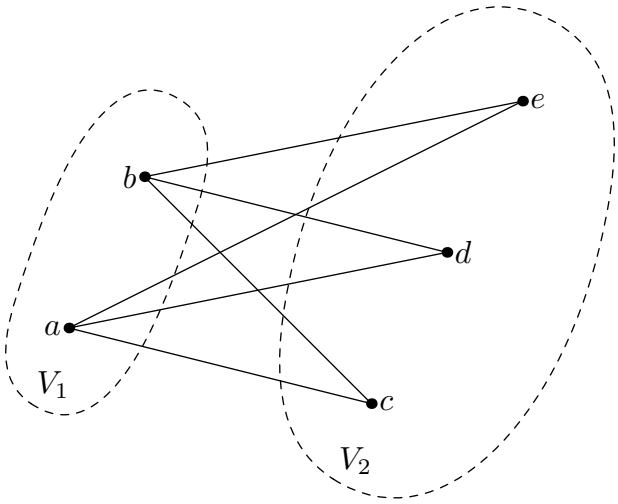
(iii) each edge in E is of the form $\{v, w\}$ with $v \in V_1$ and $w \in V_2$.

If in addition $\{v, w\}$ is an edge of the graph for every $v \in V_1$ and $w \in V_2$ then the graph (V, E) is said to be a *complete bipartite graph*. In the case when V_1 has p elements and V_2 has q elements, such a complete bipartite graph is denoted by $K_{p,q}$.

Example Let (V, E) be a graph with

$$\begin{aligned} V &= \{a,\, b,\, c,\, d,\, e\}, \\ E &= \{a\,c, a\,d, a\,e, b\,c, b\,d,\, b\,e\}. \end{aligned}$$

Let $V_1 = \{a,\, b\}$ and $V_2 = \{c,\, d,\, e\}$. Then the conditions in the above definition are satisfied by the graph (V, E) and the subsets V_1 and V_2, and therefore the graph is bipartite. Moreover it is a complete bipartite graph.



3.6 Isomorphism of Graphs

Definition An *isomorphism* between two graphs (V, E) and (V', E') is a bijective function $\varphi: V \to V'$ with the following property: for any two distinct vertices a and b belonging to V, $\{a, b\} \in E$ if and only if $\{\varphi(a), \varphi(b)\} \in E'$. If there exists such an isomorphism $\varphi: V \to V'$ between two graphs (V, E) and (V', E') then these graphs are said to be *isomorphic*.

We recall that a function $\varphi: V \to V'$ is *bijective* if and only if it has a well-defined inverse $\varphi^{-1}: V' \to V$. Thus a bijection $\varphi: V \to V'$ sets up a one-to-one correspondence between the vertices of V and V': to every vertex of V there corresponds a vertex of V', and vice versa. Such a one-to-one correspondence between the vertices belonging to V and V' is an isomorphism between the graphs (V, E) and (V', E') when it has the following additional property: a pair of distinct vertices belonging to V are the endpoints of an edge of (V, E) if and only if the corresponding vertices belonging to V' are the endpoints of an edge of (V', E'). There is then a one-to-one correspondence between the edges of the two graphs, induced by the one-to-one correspondence between their vertices.

3.7 Subgraphs

Definition Let (V, E) and (V', E') be graphs. The graph (V', E') is said to be a *subgraph* of (V, E) if and only if $V' \subset V$ and $E' \subset E$ (i.e., if and only if the vertices and edges of (V', E') are all vertices and edges of (V, E)).

Let (V, E) be a graph, and let V' be a subset of V. Let

$$E' = \{\{a, b\} \in E : a \in V' \text{ and } b \in V'\},$$

(so that E' be the set of all edges $\{a, b\}$ belonging to E whose endpoints a and b belong to V'). Then (V', E') is a subgraph of (V, E). It is referred to as the *restriction* of the graph (V, E) to V', or as the graph *induced* on V' by the graph (V, E). If the graph (V, E) is denoted by G, then its restriction (V', E') to V' may be denoted by $G|_{V'}$.

3.8 Vertex Degrees

Definition Let (V, E) be a graph. The *degree* $\deg v$ of a vertex v of this graph is defined to be the number of edges of the graph that are incident to v (i.e., the number of edges of the graph which have v as one of their endpoints).

Definition A vertex of a graph of degree 0 is said to be an *isolated* vertex.

Definition A vertex of a graph of degree 1 is said to be an *pendant* vertex.

Theorem 3.1 *Let (V, E) be a graph. Then*

$$\sum_{v \in V} \deg v = 2|E|,$$

where $\sum_{v \in V} \deg v$ denotes the sum of the degrees of all the vertices of the graph, and $|E|$ denotes the number of edges of the graph.

Proof Clearly $\sum_{v \in V} \deg v$ counts the number of times an edge of a graph is incident on a vertex of the graph. But this quantity must be twice the number of edges of the graph, since each edge is incident on exactly two vertices. ∎

Corollary 3.2 $\sum_{v \in V} \deg v$ *is an even integer.*

Corollary 3.3 *In any graph, the number of vertices of odd degree must be even.*

Definition A graph is said to be *k-regular*, for some non-negative integer k, if every vertex of the graph has degree equal to k. A *regular graph* is a graph that is k-regular for some non-negative integer k.

Corollary 3.4 *Let (V, E) be a k-regular graph. Then $k|V| = 2|E|$, where $|V|$ denotes the number of vertices and $|E|$ denotes the number of edges of the graph.*

Proof If the graph is k-regular then the sum of the degrees of the vertices of the graph is equal to $k|V|$. The result then follows immediately from Theorem 3.1. ∎

Example The graph consisting of the vertices and edges of a square is 2-regular, since every vertex (i.e., every corner of the square) is incident to exactly two edges.

Example A complete graph with n vertices is $(n-1)$-regular, since each vertex is adjacent to all the remaining $n-1$ vertices.

Example A complete bipartite graph $K_{p,q}$ is regular if and only if $p = q$.

3.9 Walks, Trails and Paths

Definition Let (V, E) be a graph. A *walk* $v_0\, v_1\, v_2\, \ldots\, v_n$ of length n in the graph from a vertex a to a vertex b is determined by a finite sequence $v_0, v_1, v_2, \ldots, v_n$ of vertices of the graph such that $v_0 = a$, $v_n = b$ and $v_{i-1}\, v_i$ is an edge of the graph for $i = 1, 2, \ldots, n$.

A walk $v_0\, v_1\, v_2\, \ldots\, v_n$ in a graph is said to *traverse* the edges $v_{i-1}v_i$ for $i = 1, 2, \ldots, n$ and to *pass through* the vertices $v_0, v_1, \ldots, v_n$.

Each vertex v in a graph determines a walk of length of length zero in the graph, consisting of the single vertex v; such a walk is said to be *trivial*.

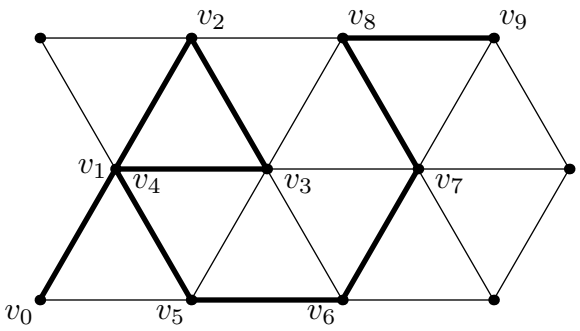
Definition Let (V, E) be a graph. A *trail* $v_0\, v_1\, v_2\, \ldots\, v_n$ of length n in the graph from a vertex a to a vertex b is a walk of length n from a to b with the property that the edges $v_{i-1}v_i$ are distinct for $i = 1, 2, \ldots, n$.

A trail in a graph is thus a walk in the graph which traverses edges of the graph at most once.

Definition Let (V, E) be a graph. A *path* $v_0\, v_1\, v_2\, \ldots\, v_n$ of length n in the graph from a vertex a to a vertex b is a walk of length n from a to b with the property that the vertices $v_0, v_1, \ldots, v_n$ are distinct.

A path in a graph is thus a walk in the graph which passes through vertices of the graph at most once.

Definition A walk, trail or path in a graph is said to be *trivial* if it is a walk v of length zero determined by a single vertex v of v; otherwise it is said to be non-trivial.



A trail $v_0\, v_1\, \ldots\, v_9$ in a graph

3.10 Connected Graphs

Definition An undirected graph is said to be *connected* if, given any two vertices u and v of the graph, there exists a path in the graph from u to v.

Theorem 3.5 *Let u and v be vertices of a graph. Then there exists a path in the graph from u to v if and only if there exists a walk in the graph from u to v.*

Proof Any path in a graph from one vertex to another is a walk. It therefore only remains to show that if there exists a walk in the graph from a vertex u to a vertex v, then there must also exist a path in the graph from u to v.

Now if there exists at least one walk from u to v, then there must exist a walk from u to v whose length is less than or equal to that of every other walk from u to v. Let this walk be $a_0\, a_1\, \ldots\, a_n$, where $a_0 = u$ and $a_n = v$. We claim that this walk is in fact a path from u to v. Indeed were it the case that $a_j = a_k$ for some integers j and k satisfying $0 \leq j < k \leq n$ then the walk $a_0\, \ldots\, a_j\, a_{k+1}\, \ldots\, a_n$ from u to v obtained on omitting the edges $a_j\, a_{j+1}, \ldots, a_{k-1}\, a_k$ would be a walk from u to v whose length was strictly less than that of the given walk (which is the *shortest* walk from u to v). But this is clearly impossible. Hence $a_0, a_1, \ldots, a_n$ must be distinct, and thus the the walk $a_0\, a_1\, \ldots\, a_n$ is a path from u to v. ∎

Corollary 3.6 *An undirected graph is connected if and only if, given any two vertices u and v of the graph, there exists a walk in the graph from u to v.*

3.11 The Components of a Graph

Let (V, E) be an undirected graph. We can define a relation $\sim$ on the set V of vertices of the graph, where two vertices a and b of the graph satisfy $a \sim b$

if and only if there exists a walk in the graph from a to b.

Lemma 3.7 *Let (V, E) be an undirected graph. Then the relation $\sim$ on the set V of vertices of the graph is an equivalence relation, where two vertices u and v of the graph satisfy $u \sim v$ if and only if there exists a walk in the graph from u to v.*

Proof We must prove that the relation $\sim$ on V is reflexive, symmetric and transitive.

Clearly $v \sim v$ for any vertex v of the graph, since the trivial walk v is walk from v to itself. Thus the relation $\sim$ is reflexive.

Let u and v be vertices of the graph satisfying $u \sim v$. Then there exists a walk $u\, a_1\, a_2\, \ldots\, a_{n-1}\, v$ from u to v. This walk may be reversed to obtain a walk $v\, a_{n-1}\, a_{n-2}\, \ldots\, a_1\, u$ from v to u. We conclude that if $u \sim v$ then $v \sim u$. Thus the relation $\sim$ is symmetric.

Finally let u, v and w be vertices of the graph for which $u \sim v$ and $v \sim w$. Then there exists a walk $u\, a_1\, a_2\, \ldots\, a_{n-1}\, v$ from u to v, and a walk $v\, b_1\, b_2\, \ldots\, b_{r-1}\, w$ from v to b. These two walks may be concatenated to yield a walk

$$u\, a_1\, a_2\, \ldots\, a_{n-1}\, v\, b_1\, b_2\, \ldots\, b_{r-1}\, w$$

from u to w, showing that $u \sim w$. Thus the relation $\sim$ is transitive. We have shown that this relation is reflexive, symmetric and transitive. It is therefore an equivalence relation. ∎

The equivalence relation $\sim$ on the set V of vertices of the graph (V, E) gives rise to a partition of V as the disjoint union of subsets $V_1, V_2, \ldots, V_m$, where

(i) $V_1 \cup V_2 \cup \cdots \cup V_k = V$;

(ii) $V_i \cap V_j = \emptyset$ if $i \neq j$;

(iii) two vertices u and v belong to a single subset V_i if and only if there exists a walk in (V, E) from u to v (i.e., if and only if $u \sim v$).

If u and v are the endpoints of some edge $u\, v$ of the graph (V, E), then $u \sim v$ (since an edge can be considered as a walk of length one), and thus u and v belong to the same set V_i. Thus, if we define

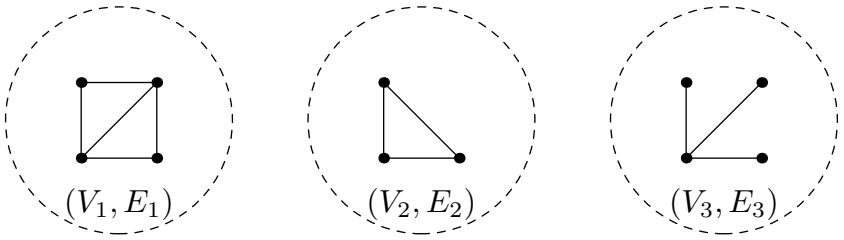
$$E_i = \{u\, v \in E : u \in V_i \text{ and } v \in V_i\},$$

then (V_1, E_1), $(V_2, E_2), \ldots,$ (V_k, E_k) are subgraphs of (V, E), and

$$V = V_1 \cup V_2 \cup \cdots \cup V_k, \quad E = E_1 \cup E_2 \cup \cdots \cup E_k.$$

These subgraphs are *disjoint* since $V_i \cap V_j = \emptyset$ and $E_i \cap E_j = \emptyset$ if $i \neq j$. Moreover the graph (V_i, E_i) is the restriction of the graph (V, E) to V_i (also describable as the graph induced on V_i by (V, E)) for $i = 1, 2, \ldots, k$.

The subgraphs (V_i, E_i) of (V, E) are referred to as the *components* (or *connected components*) of the graph (V, E).



A graph with three components

Lemma 3.8 *The vertices and edges of any walk in an undirected graph are all contained in a single component of that graph.*

Proof Let $v_0\, v_1 \ldots v_n$ be a walk in a graph (V, E). Then $v_0\, v_1 \ldots v_r$ is a walk in (V, E) from v_0 to v_r for each integer r between 1 and m. It follows that each vertex v_r through which the walk passes must belong to the same component of the graph as v_0. Therefore all the vertices and edges of this walk belong to a single component of the graph, namely that component which contains the vertex v_0. ∎

Lemma 3.9 *Each component of an undirected graph is connected.*

Proof Let (V, E) be a graph, and let u and v be vertices belonging to V_i, where (V_i, E_i) is one of the components of this graph. Then there exists a walk in (V, E) from u to v. But the vertices and edges of this walk are contained in a single component of the graph (V, E), by Lemma 3.8, and that component must obviously be the component (V_i, E_i) that contains the vertices u and v. Thus there exists a walk in (V_i, E_i) from u to v. We conclude that the graph (V_i, E_i) is connected. ∎

Remark The importance of the concept of the components of a graph is that it enables us to reduce the study of undirected graphs in general to the study of connected graphs. Indeed any undirected graph can be represented as a disjoint union of connected subgraphs: these subgraphs are the components of the given graph. These connected components may then be studied

individually. Moreover properties of any one component do not affect those of any other, since no edge of the graph passes from any one component of the graph to any other.

3.12 Circuits

Definition Let (V, E) be a graph. A *walk* $v_0\, v_1\, v_2\, \ldots\, v_n$ in the graph is said to be *closed* if $v_0 = v_n$.

Thus a walk in a graph is closed if and only if it starts and ends at the same vertex.

Definition Let (V, E) be a graph. A *circuit* in the graph is a non-trivial closed trail in the graph.

We see therefore that a circuit in a graph is a closed walk with no repeated edges, and passing though at least two vertices.

Definition A circuit $v_0\, v_1\, v_2\, \ldots\, v_{n-1}\, v_0$ in a graph is said to be *simple* if the vertices $v_0, v_1, v_2, \ldots\, v_{n-1}$ are distinct.

Remark Some authors use the term *cycle* to denote a simple circuit in a graph. Others use the term *cycle* to refer to a circuit in the graph, irrespective of whether or not it is simple.

We now prove two theorems that provide sufficient conditions for a graph to contain simple circuits.

Theorem 3.10 *If a graph has no isolated or pendant vertices then it contains at least one simple circuit.*

Proof Let (V, E) be a graph with no isolated or pendant vertices. The length of any path in this graph cannot exceed $|V| - 1$, where $|V|$ denotes the number of vertices of the graph, since a path of length m passes through $m + 1$ distinct vertices. Therefore there exists a path $v_0\, v_1\, v_2\, \ldots\, v_m$ in the graph whose length m is greater than or equal to the length of every other path in the graph. Now the final vertex v_m of the graph is adjacent to at least two vertices of the graph, since the graph contains no isolated or pendant vertices. One of these vertices is v_{m-1}. If none of the vertices $v_0, v_1, \ldots, v_{m-2}$ were incident to v_m then there would exist a vertex w adjacent to v_m that was distinct from $v_0, v_1, \ldots, v_m$, and then $v_0, v_1, \ldots, v_m\, w$ would be a path in the graph with length exceeding m, which is impossible. It follows that at

least one of the vertices $v_0, v_1, \ldots, v_{m-2}$ is incident to v_m; let that vertex be v_k, where $0 \leq k \leq m-2$. Then $v_k\, v_{k+1} \ldots v_m\, v_k$ is a simple circuit in the graph. Thus a graph with no isolated or pendant vertices always contains a simple circuit. ∎

Theorem 3.11 *Let u and v be vertices of a graph, where $u \neq v$. Suppose that there exist at least two distinct paths in the graph from u to v. Then the graph contains at least one simple circuit.*

Proof Let $a_0\, a_1\, a_2 \ldots a_m$ and $b_0\, b_1\, b_2 \ldots b_n$ be two distinct paths in the graph with $a_0 = b_0 = u$ and $a_m = b_n = v$. We may suppose that $m \leq n$. Now the fact that paths are distinct ensures that there exists at least integer i satisfying $0 < i \leq m$ for which $a_i \neq b_i$. Let the smallest such integer i be $r+1$, where r is an integer in the range $0 \leq r < m$. Then $a_r = b_r$ and $a_{r+1} \neq b_{r+1}$. Now the condition $a_i \in \{b_j : r < j \leq n\}$ is satisfied when $i = m$, since $a_m = b_n$. Let s be the smallest integer satisfying $r < s \leq m$ for which $a_s \in \{b_j : r < j \leq n\}$. Then $a_s = b_t$ for some integer t satisfying $r < t \leq n$. Moreover none of the vertices a_i with $r < i < s$ belong to the set $\{b_j : r < j < t\}$. It follows that

$$a_r\, a_{r+1} \ldots a_s\, b_{t-1} \ldots b_{r+1} a_r$$

is a simple circuit in the graph. Thus the graph has at least one simple circuit, as required. ∎

3.13 Eulerian Trails and Circuits

Definition An *Eulerian trail* in a graph is a trail that traverses every edge of the graph.

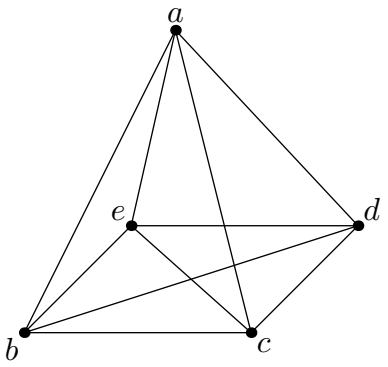
Note that an Eulerian trail in a graph must traverse every edge of the graph exactly once, since a trail traverses an edge of the graph at most once.

Definition An *Eulerian circuit* in a graph is a circuit that traverses every edge of the graph.

It follows from these definitions that any closed Eulerian trail is an Eulerian circuit.

Example Let (V, E) be the complete graph K_5 on five vertices a, b, c, d and e, where

$$\begin{aligned} V &= \{a, b, c, d, e\}, \\ E &= \{a\,b, a\,c, a\,d, a\,e, b\,c, b\,d, b\,e, c\,d, c\,e, d\,e\}. \end{aligned}$$



This graph has Eulerian circuits. One of them is the following:

$$a\,b\,c\,a\,d\,e\,c\,d\,b\,e\,a.$$

Remark Eulerian trails and circuits are named after the Swiss mathematician Leonhard Euler (1707–1783), who first studied the problem of the existence of such circuits in connection with the problem of the Seven Bridges of Königsberg. The citizens of this city used to amuse themselves by attempting to devise a walk around the city that would cross each of the seven bridges exactly once. They always failed in this attempt, for reasons explained by Euler.

We shall derive necessary and sufficient conditions for the existence of Eulerian trails and circuits in a connected graph. The following theorem will give rise to a necessary condition for the existence of an Eulerian trail or circuit.

Theorem 3.12 *Let $v_0\,v_1\,\ldots\,v_m$ be a trail in a graph, and let v be a vertex of that graph. Then the number of edges of the trail incident to the vertex v is even, except in the case when the trail is not closed and the trail starts or finishes at v, in which case the number of edges of the trail incident to the vertex v is odd.*

Proof First suppose that $v \neq v_0$ and $v \neq v_m$. The edges of the trail that are incident to v are then those of the form $v_{i-1}\,v_i$ and $v_i\,v_{i+1}$ with $0 < i < m$ and $v_i = v$. It follows that the number of edges of the trail incident to v is then equal to twice the number of integers i satisfying $0 < i < m$ for which $v = v_i$, and is thus even.

If $v = v_0$, and if the trail is not closed (i.e., if $v_m \neq v_0$), then the edges of the trail incident to v are the edge $v_0\,v_1$ together with the edges $v_{i-1}\,v_i$ and $v_i\,v_{i+1}$ for those integers i satisfying $1 < i < m$ for which $v = v_i$. Therefore the number of edges of the trail incident to v is then equal to one plus twice the number of integers i satisfying $1 < i < m$ for which $v = v_i$, and is thus odd. Similarly the number of edges of the trail incident to v is odd in the case when $v = v_m$ and the trail is not closed. Finally, in the case when the trail is closed and $v = v_0 = v_m$, the edges incident to v are $v_0\,v_1$ and $v_{m-1}\,v_m$, together with the edges $v_{i-1}\,v_i$ and $v_i\,v_{i+1}$ for those integers i satisfying $1 < i < m$ for which $v = v_i$. The total number of edges of the trail incident to v is therefore even. ∎

Corollary 3.13 *Let v be a vertex of a graph. Then, given any circuit in the graph, the number of edges incident to v that are traversed by that circuit is even.*

Corollary 3.14 *If a graph admits an Eulerian circuit then the degree of every vertex of the graph must be even.*

Proof Let v be a vertex of the graph. It follows from Corollary 3.13 that the number of edges of any Eulerian circuit incident to v is even. But every edge incident to v is an edge of an Eulerian circuit, since an Eulerian circuit by definition traverses every edge of the graph. It follows that the degree of the vertex v is even, as required. ∎

Example Any attempt to find an Eulerian circuit in the complete graph K_4 on four vertices is guaranteed to fail, since such a graph is 3-regular (i.e., the degree of each of the four vertices of the graph is equal to 3).

Corollary 3.15 *If a graph admits an Eulerian trail that is not a circuit then the degrees of exactly two vertices of the graph must be odd, and the degrees of the remaining vertices must be even. The two vertices with odd degrees will then be the initial and final vertices of the Eulerian trail.*

Proof As in the proof of Corollary 3.15 we see from Theorem 3.12 that the degree of a vertex of the graph must be even unless that vertex is one of the two endpoints of the trail, in which case the degree must be odd. ∎

We shall now work towards a proof of the fact that a non-trivial connected graph has an Eulerian circuit if the degree of each of its vertices is even. For this we use the results of the following lemmas.

Lemma 3.16 *Let $v\,w$ be an edge of a graph in which the degree of every vertex is even. Then there exists a circuit of the graph which traverses the edge $v\,w$.*

Proof Let $v_0 = v$ and $v_1 = w$. Suppose that, for some positive integer k a trail $v_0\,v_1\,\ldots\,v_k$ has been constructed in the graph starting at the vertex v and traversing the edge $v\,w$. Suppose also that $v_k \neq v$. It follows from Theorem 3.12 that the number of edges of the trail incident to v_k must be odd. But the degree of v_k is even. It follows that the number of edges of the trail incident to v_k must be strictly less than the degree of v_k, and therefore there must exist at least one edge of the graph incident to v_k which is not traversed by the trail $v_0\,v_1\,\ldots\,v_k$. Let that edge be $v_k\,v_{k+1}$, where v_{k+1} is a vertex adjacent to v_k. Then $v_0\,v_1\,\ldots\,v_k\,v_{k+1}$ is a trail of length $k+1$ in the graph which starts at v and traverses the edge $v\,w$.

Now the length of any trail in a graph cannot exceed the number of edges of the graph, since each edge of the graph is traversed at most once by any trail. It follows that successive extensions of the trail $v\,w$ will ultimately result in a trail that cannot be extended to a longer trail. This must then be closed (since we have just shown that if the trail is not closed then it can always be extended). This closed trail is then the required circuit. ∎

Lemma 3.17 *Suppose that a graph contains a circuit of length m and a circuit of length n. Suppose also that no edge of the graph is traversed by both circuits, and that at least one vertex of the graph is common to both circuits. Then the graph contains a circuit of length $m+n$.*

Proof Let u be a vertex of the graph which is common to both circuits. We may clearly suppose that both circuits start from and finish at this vertex u. Let the first circuit be $u\,v_1\,\ldots\,v_{m-1}\,u$ and let the second circuit be $u\,w_1\,\ldots\,w_{n-1}\,u$. We can then concatenate these two circuits together to obtain a third circuit

$$u\,v_1\,\ldots\,v_{m-1}\,u\,w_1\,\ldots\,w_{n-1}u$$

of length $m+n$. ∎

Lemma 3.18 *Let (V, E) be a connected graph, and let some trail in this graph be given. Suppose that no vertex of the graph has the property that some but not all of the edges of the graph incident to that vertex are traversed by the trail. Then the given trail is an Eulerian trail.*

Proof Let V_1 be the set of vertices through which the trail passes, and let V_2 denote the set consisting of any remaining vertices of the graph. Then $V_1 \cup V_2 = V$ and $V_1 \cap V_2 = \emptyset$. We will prove that $V_2 = \emptyset$.

Now any vertex belonging to V_1 is incident to at least one edge traversed by the trail. But then all edges incident to a vertex belonging to V_1 must be traversed by the trail. But then any vertex of V adjacent to a vertex in V_1 must itself belong to V_1, and thus no edge can join a vertex in V_1 to a vertex in V_2. If the set V_2 were non-empty then there could not exist any path joining a vertex in V_2 to a vertex in V_1, and thus the graph would not be connected. Therefore V_2 must be empty, and $V_1 = V$. But then every edge of (V, E) must be traversed by the trail, and thus the trail is an Eulerian trail. ∎

Lemma 3.19 *Let (V, E) be a connected graph with the property that the degree of every vertex of the graph is even, and let some circuit in this graph be given. Suppose that there is some vertex v of the graph with the property that some but not all of the edges of the graph incident to that vertex are traversed by the given circuit. Then there exists a second circuit in the graph (V, E) which passes through the vertex v and which does not traverse any edge which is traversed by the given circuit.*

Proof let E' denote the subset of E consisting of those edges of the graph that are not traversed by the given circuit. Then (V, E') is a subgraph of the given graph (V, E). Given any vertex w, the number of edges of the given circuit that are incident to w is equal to $d(w) - d'(w)$, where $d(w)$ is the number of edges in E incident to w, and $d'(w)$ is the number of edges in E' incident to w. It follows from Lemma 3.13 that $d(w) - d'(w)$ is an even integer. But the degree $d(w)$ of each vertex w of the graph (V, E) is even, by assumption, and therefore $d'(w)$ is also even. Thus the degree of every vertex in the subgraph (V, E') is even.

Now the vertex v of the graph has the property that some but not all of edges incident to this vertex are traversed by the given trail. Therefore some at least of the edges of the graph (V, E) incident to v are edges also of the subgraph (V, E'). It then follows from Lemma 3.16 that the subgraph (V, E') contains a circuit which passes through the vertex v. This circuit is of course a circuit in the graph (V, E), it passes through the vertex v, and it does not traverse any edge of the graph (V, E) that is traversed by the given circuit. ∎

Theorem 3.20 *A non-trivial connected graph contains an Eulerian circuit if the degree of every vertex of the graph is even.*

Proof Let (V, E) be a non-trivial connected graph with the property that the degree of every vertex is even. An easy application of Lemma 3.16 shows

that such a graph contains at least one circuit. It therefore contains a circuit which is at least as long as every other circuit in the graph. We shall show that this circuit of maximal length is an Eulerian circuit.

Now if the graph were to contain some vertex v with the property that some but not all of the edges of the graph incident to that vertex are traversed by this circuit of maximal length, then it would follow from Lemma 3.19 that there would exist a second circuit in the graph (V, E) which would also pass through the vertex v, and which would not traverse any edge traversed by the circuit of maximal length. But it would then follow immediately from Lemma 3.17 that the graph would contain a circuit which was longer than the circuit of maximal length, which is clearly impossible.

We conclude therefore that the graph cannot contain any vertex v with the property that some but not all of the edges of the graph incident to that vertex are traversed by the circuit of maximal length. It now follows from Lemma 3.18 that such a circuit of maximal length must be an Eulerian circuit. ∎

Remark A careful examination of the proofs of Lemma 3.16, Corollary 3.17 and Lemma 3.19 shows that they provide an algorithm for constructing an Eulerian circuit in a non-trivial connected graph whose vertices all have even degree. Indeed the proof of Lemma 3.16 shows how circuits can be constructed in such a graph, and the proofs of Corollary 3.17 and Lemma 3.19 show how to replace a circuit that is not an Eulerian circuit by a strictly longer circuit. A finite number of such replacements must ultimately result in an Eulerian circuit.

On combining the results of Corollary 3.14 and Theorem 3.20 we conclude that a non-trivial connected graph has an Eulerian circuit if and only if the degree of each of its vertices is even.

We now prove the result corresponding to Theorem 3.20 for non-trivial connected graphs with exactly two vertices whose degree is odd.

Corollary 3.21 *Suppose that a connected graph has exactly two vertices whose degrees are odd. Then there exists an Euler trail in the graph joining the two vertices with odd degrees.*

Proof Let (V, E) be the graph, and let v and w be the two vertices of this graph whose degree is odd. We may embed the graph (V, E) as a subgraph of a larger graph (V', E') whose vertices all have even degree. We choose the graph (V', E') such that $V' = V \cup \{u\}$ and $E' = E \cup \{v\,u, u\,w\}$, where u is a vertex of V' that does not belong to V, and is the only such vertex of

V'. The graph (V', E') is then non-trivial and connected, and every vertex of (V', E') has even degree. (Indeed the degree of the vertex u in the graph (V', E') is equal to 2, and the degrees of the vertices v and w in the graph (V', E') exceed by one their degrees in the graph (V, E).) It follows from Theorem 3.20 that the graph (V', E') has an Eulerian circuit. We may order the vertices of this circuit so that the final two edges of the circuit are $w\,u$ and $u\,v$. Deletion of these two edges from the circuit yields the required Eulerian trail in the graph (V, W) from v to w. ∎

3.14 Hamiltonian Paths and Circuits

Definition A *Hamiltonian path* in a graph is a path that passes (exactly once) through every vertex of the graph.

Thus a path $v_0\,v_1\,v_2\,\ldots\,v_n$ in a graph (V, E) is a Hamiltonian path if and only if $V = \{v_0, v_1, \ldots, v_n\}$. A Hamiltonian path passes can have no repeated vertices (since it is a path) and therefore passes through each vertex of the graph exactly once.

Definition A *Hamiltonian circuit* in a graph is a simple circuit that passes through every vertex of the graph.

Thus a circuit $v_0\,v_1\,v_2\,\ldots\,v_{n-1}\,v_0$ in a graph (V, E) is a Hamiltonian circuit if and only if every vertex of the graph occurs exactly once in the list $v_0, v_1, \ldots, v_{n-1}$.

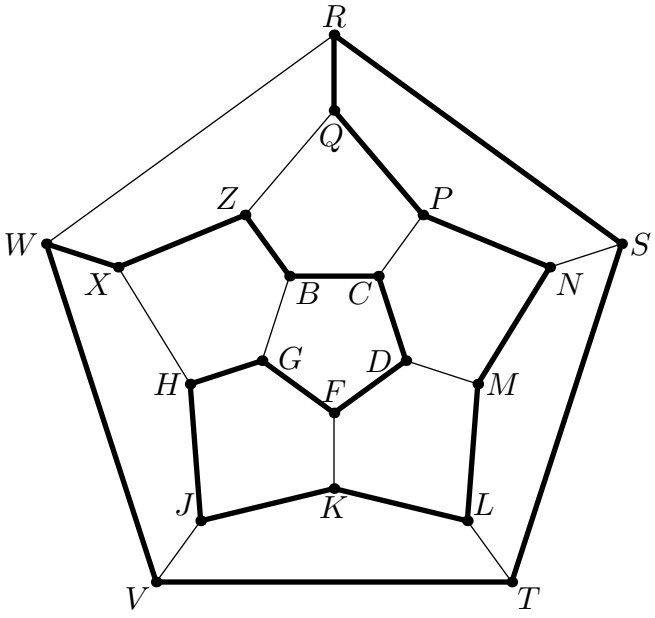
Remark Hamiltonian circuits are are named after William Rowan Hamilton (1805–1865), who showed in 1856 that such circuits could be found in the graph consisting of the vertices and edges of a dodecahedron. Hamilton developed an 'icosian calculus' for the study such circuits in the dodecahedron, and formulated a game, the *icosian game*, in which people were challenged to complete any path of length two in this graph to a Hamilton circuit in the graph.

3.15 Forests and Trees

Definition A graph is said to be *acyclic* if it contains no circuits.

Definition A *forest* is an acyclic graph.

Definition A *tree* is a connected forest.



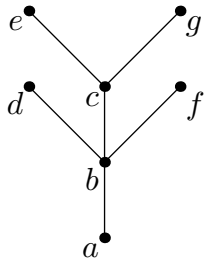
Hamilton's circuit round the edges of a dodecahedron

Note that the components of any forest are trees.

Example The graph (V, E), where

$$\begin{aligned} V &= \{a,\, b,\, c,\, d,\, e,\, f,\, g\}, \\ E &= \{a\,b,\, b\,c,\, b\,d,\, c\,e,\, b\,f,\, c\,g\}, \end{aligned}$$

is a tree.



The vertices a, d, e, f and g are pendant vertices (i.e., each of these vertices is incident to exactly one edge of the graph, and is therefore of degree one.) The tree has 7 vertices and 6 edges.

Theorem 3.22 *Every forest contains at least one isolated or pendant vertex.*

Proof If a graph has no isolated or pendant vertices, then it contains a circuit (Theorem 3.10). But a forest contains no circuits. Therefore must have at least one isolated or pendant vertex. ∎

Theorem 3.23 *A non-trivial tree contains at least one pendant vertex.*

Proof A non-trivial graph has more than one vertex. If a non-trivial graph has an isolated vertex then there does not exist any path or walk from that vertex to any other vertex of the graph, and therefore the graph is not connected. But a tree is by definition connected. Therefore a non-trivial tree cannot have any isolated vertex. However a tree is a forest, and therefore contains at least one vertex that is either an isolated vertex or a pendant vertex (Theorem 3.22). Such a vertex must then be a pendant vertex. ∎

Theorem 3.24 *Let (V, E) be a tree. Then $|E| = |V| - 1$, where $|V|$ and $|E|$ denote respectively the number of vertices and the number of edges of the tree.*

Proof We can prove the result by induction on the number $|V|$ of vertices of the tree. The result is clearly true when the tree is trivial, since it then consists of one vertex and no edges.

Suppose that every tree with m vertices has $m - 1$ edges. Let (V, E) be a tree with $m + 1$ vertices. At least of these vertices is a pendant vertex (Theorem 3.23). Let v be a pendent vertex, let w be the vertex that is adjacent to v, let $V' = V \setminus \{v\}$, and let $E' = E \setminus \{v\,w\}$. Then (V', E') is a subgraph of (V, E), and this subgraph has m vertices. (This subgraph is obtained from the original graph by deleting the vertex v and the edge $v\,w$ from that graph.) We claim that this subgraph (V', E') is in fact a tree.

First we show that (V', E') is connected. Now, given any two vertices in V', there exists a path in (V, E) from one vertex to the other. This path could not pass through the vertex v, since otherwise the path would have to pass through w twice (going out to v and then returning from v), which is impossible since a path by definition has no repeated vertices. Therefore this path is in fact a path in (V', E'). We conclude that (V', E') is connected.

Now the tree (V, E) does not contain any circuits. It follows immediately that the connected subgraph (V', E') does not contain any circuits, and is thus a tree. It has m vertices.

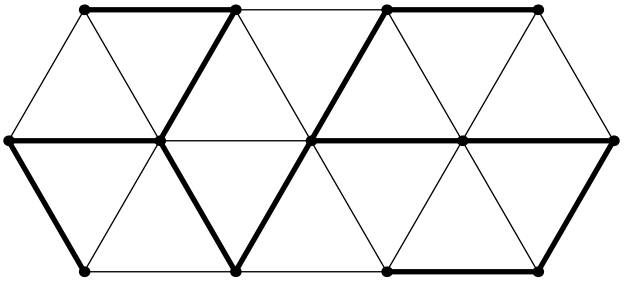
The induction hypothesis now ensures that the tree (V', E') has $m - 1$ edges, and therefore the tree (V, E) has m edges. The required result therefore follows by the Principle of Mathematical Induction. ∎

Theorem 3.25 *Given two distinct vertices of a tree, there exists a unique path in the tree from the first vertex to the the second.*

Proof Let u and v be distinct vertices of the tree. There must exist at least one path in the tree from u to v, since any tree is connected. Were there to exist more than one, then it would follow from Theorem 3.11 that there would exist at least one circuit in the tree, which is impossible, since that a tree cannot contain any circuits. Therefore there must exist exactly one path in the tree from u to v. ∎

3.16 Spanning Trees

Definition A *spanning tree* in a graph (V, E) is a subgraph of the graph (V, E) that is a tree which includes every vertex of the graph (V, E).



A spanning tree in a graph

Theorem 3.26 *Every connected graph contains a spanning tree*

Proof Let (V, E) be a connected graph. The collection consisting of all the connected subgraphs of (V, E) with the same vertices as (V, E) is non-empty, since it includes the graph (V, E) itself. Choose a subgraph (V, E') in this collection such that the number $|E'|$ of edges in this subgraph is less than or equal to the number of edges of any other subgraph in the collection. We claim that (V, E') is the required spanning tree. Clearly (V, E') is connected and has the same vertices as V. It only remains to show that (V, E') does not contain any circuits.

Suppose that (V, E') were to contain a circuit. Let $v\,w$ be an edge traversed by some circuit in (V, E'), and let $E'' = E \setminus \{v\,w\}$. There would then exist a walk from v to w whose edges belong to E''. (Such a walk could

consist of the remaining edges of the circuit traversing the edge $v\,w$.) Moreover every vertex in V could be joined to v by a walk whose edges belong to E', and could therefore be joined either to v or to w by a walk whose edges belong to E''. It would then follow that every vertex of V could be joined to v by a walk whose edges belong to E'', and therefore the graph (V, E'') would be a connected subgraph of (V, E) with the same vertices as (V, E) and with fewer edges than (V, E'), which is impossible. We conclude therefore that the subgraph (V, E') of (V, E) cannot contain any circuits and is therefore the required spanning tree. ∎

Corollary 3.27 *Let (V, E) be a connected graph with $|V|$ vertices and $|E|$ edges. Suppose that $|E| = |V| - 1$. Then the graph (V, E) is a tree.*

Proof A connected graph (V, E) contains a spanning tree, by Theorem 3.26. This spanning tree must have $|V| - 1$ edges, by Theorem 3.24. But the spanning tree then has the same number of edges as the original graph (V, E), and must therefore be the same as this graph. It follows that the graph (V, E) must be a tree, since it is a spanning tree of itself. ∎

3.17 Directed Graphs

Definition An *directed graph* or *digraph* (V, E) consists of a finite set V together with a subset E of $V \times V$. The elements of V are the *vertices* of the digraph; the elements of E are the *edges* of the digraph.

An edge of a digraph (V, E) is an ordered pair (a, b) where a and b are vertices of the graph. These vertices need not be distinct: a digraph may contain *loops* of the form (a, a), where a is some vertex of the digraph. Also the vertices of an edge of a digraph are ordered: if a and b are distinct vertices of the graph then $(a, b) \neq (b, a)$, and moreover neither, one only, or both of (a, b) and (b, a) may be edges of the digraph.

Let (a, b) be an edge of a directed graph (V, E). We say that a is the *initial vertex* and b is the *terminal vertex* of the edge. Moreover we say that the vertex b is *adjacent from* the vertex a, and the vertex a is *adjacent to* the vertex b, and the edge (a, b) is *incident from* the vertex a and *incident to* the vertex b.

3.18 Adjacency Matrices of Directed Graphs

Definition Let (V, E) be a directed graph, and let the vertices of the graph be ordered as $v_1, v_2, \ldots, v_m$. The *adjacency matrix* of the directed graph is

the $m \times m$ matrix (b_{ij}), or

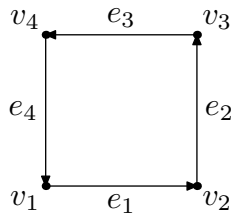
$$\begin{pmatrix} b_{11} & b_{12} & \dots & b_{1m} \\ b_{21} & b_{22} & \dots & b_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & a_{m2} & \dots & b_{mm} \end{pmatrix},$$

where

$$b_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \in E; \\ 0 & \text{otherwise.} \end{cases}$$

Example Let (V, E) be the directed graph whose vertices are ordered as v_1, v_2, v_3 and v_4, and whose edges are ordered as e_1, e_2, e_3 and e_4, where

$$e_1 = (v_1, v_2), \quad e_2 = (v_2, v_3), \quad e_3 = (v_3, v_4), \quad e_4 = (v_4, v_1).$$



The adjacency matrix of this digraph is

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix},$$

3.19 Directed Graphs and Binary Relations

There is a correspondence between directed graphs and binary relations on finite sets.

Let V be a finite set. Corresponding to any relation R on V there is a directed graph (V, E), where

$$E = \{(a, b) \in V \times V : aRb\}.$$

Conversely any directed graph (V, E) gives rise to a relation R on the set V of vertices of the digraph, where vertices a and b of the graph satisfy aRy if and only if $(a, b) \in R$.

4 Abstract Algebra

4.1 Binary Operations on Sets

Definition A *binary operation* $*$ on a set A is an operation which, when applied to any elements x and y of the set A, yields an element $x * y$ of A.

Example The arithmetic operations of addition, subtraction and multiplication are binary operations on the set $\mathbb{R}$ of real numbers which, when applied to real numbers x and y, yield the real numbers $x + y$, $x - y$ and xy respectively.

However division is not a binary operation on the set of real numbers, since the quotient x/y is not defined when $y = 0$. (Under a binary operation $*$ on a set must determine an element $x * y$ of the set for every pair of elements x and y of that set.)

4.2 Commutative Binary Operations

Definition A binary operation $*$ on a set A is said to be *commutative* if $x * y = y * x$ for all elements x and y of A.

Example The operations of addition and multiplication on the set $\mathbb{R}$ of real numbers are commutative, since $x + y = y + x$ and $x \times y = y \times x$ for all real numbers x and y. However the operation of subtraction is not commutative, since $x - y \neq y - x$ in general. (Indeed the identity $x - y = y - x$ holds only when $x = y$.)

4.3 Associative Binary Operations

Let $*$ be a binary operation on a set A. Given any three elements x, y and z of a set A, the binary operation, applied to the elements $x * y$ and z of A, yields an element $(x * y) * z$ of A, and, applied to the elements x and $y * z$ of A, yields an element $x * (y * z)$ of A.

Definition A binary operation $*$ on a set A is said to be *associative* if $(x * y) * z = x * (y * z)$ for all elements x, y and z of A.

Example The operations of addition and multiplication on the set $\mathbb{R}$ of real numbers are associative, since $(x + y) + z = x + (y + z)$ and $(x \times y) \times z = x \times (y \times z)$ for all real numbers x, y and z. However the operation of subtraction is not associative. For example $(1 - 2) - 3 = -4$, but $1 - (2 - 3) = 2$.

When a binary operation $*$ is associative it is not necessary to retain the parentheses in expressions such as $(x * y) * z$ or $x * (y * z)$. These two expressions may both be written without ambiguity as $x * y * z$.

4.4 Semigroups

Definition A *semigroup* consists of a set on which is defined an associative binary operation.

We may denote by $(A, *)$ a semigroup consisting of a set A together with an associative binary operation $*$ on A.

Definition A semigroup $(A, *)$ is said to be *commutative* (or *Abelian*) if the binary operation $*$ is commutative.

Example The set of natural numbers, with the operation of addition, is a commutative semigroup, as is the set of natural numbers with the operation of multiplication.

Let $(A, *)$ be a semigroup. Given any element a of A, we define

$$\begin{aligned} a^1 &= a, \\ a^2 &= a * a, \\ a^3 &= a * a^2 = a * (a * a), \\ a^4 &= a * a^3 = a * (a * (a * a)), \\ a^5 &= a * a^4 = a * (a * (a * (a * a))), \\ &\vdots \end{aligned}$$

In general we define a^n recursively for all natural numbers n so that $a^1 = a$ and $a^n = a * a^{n-1}$ whenever $n > 1$.

Remark In the case of the semigroup consisting of the set of natural numbers with the operation of multiplication, the value of 'a^n' given by the above rule is the nth power of a natural number a. However in the case of the semigroup consisting of the set of natural numbers with the operation of addition it is not the nth power of a, but is na.

Theorem 4.1 *Let $(A, *)$ be a semigroup, and let a be an element of A. Then $a^m * a^n = a^{m+n}$ for all natural numbers m and n.*

Proof We prove this theorem by induction on m.

Now it follows immediately from the definition of a^{n+1} that $a * a^n = a^{1+n}$ for all natural numbers n. Thus the theorem is true in the case when $m = 1$.

Suppose that the required result is true in the case when $m = s$ for some natural number s, so that $a^s * a^n = a^{s+n}$ for all natural numbers n. Then

$$a^{s+1} * a^n = (a * a^s) * a^n = a * (a^s * a^n) = a * a^{s+n} = a^{s+1+n}$$

for all natural numbers n. Thus if the required result is true when $m = s$ then it is also true when $m = s + 1$. We conclude using the Principle of Mathematical Induction that the identity $a^m * a^n = a^{m+n}$ holds for all natural numbers m and n, as required. ∎

Theorem 4.2 *Let $(A, *)$ be a semigroup, and let a be an element of A. Then $(a^m)^n = a^{mn}$ for all natural numbers m and n.*

Proof The result may be proved by induction on the natural number n. The identity $(a^m)^n = a^{mn}$ clearly holds whenever $n = 1$. Suppose that s is a natural number with the property that $(a^m)^s = a^{ms}$ for all natural numbers m. Then

$$(a^m)^{s+1} = (a^m)^s * a^m = a^{ms} * a^m = a^{ms+m} = a^{m(s+1)}.$$

Thus if the identity $(a^m)^n = a^{mn}$ holds when $n = s$ then it also holds when $n = s + 1$. We conclude from the Principle of Mathematical Induction that this identity holds for all natural numbers n.

Remark Note that the above proof made use of the fact that the binary operation on a semigroup is associative.

4.5 The General Associative Law

Let $(A, *)$ be a semigroup, and let x, y, z and w be elements of A. We can use the associative property of $*$ to show that the value of a product involving x, y, z, w is independent of the manner in which that product is bracketed, though it generally depends on the order in which x, y, z and w occur in that product (unless that binary operation is also commutative). For example,

$$\begin{aligned}(x * (y * z)) * w &= ((x * y) * z) * w \\ &= (x * y) * (z * w) \\ &= x * (y * (z * w)) \\ &= x * ((y * z) * w)\end{aligned}$$

All the above products may therefore be denoted without ambiguity by the expression $x * y * z * w$ from which the parentheses have been dropped.

The analogous property holds for products involving five or more elements of the semigroup.

In any semigroup, the value of a product of three or more elements of the semigroup depends in general on the order in which those elements occur in the product (unless the binary operation is commutative), but the value of the product is independent of the manner in which the product is bracketed. This general result is often referred to as the General Associative Law, and can be proved using induction on the number of elements that occur in the product.

4.6 Identity elements

Definition Let $(A, *)$ be a semigroup. An element e of A is said to be an *identity element* for the binary operation $*$ if $e*x = x*e = x$ for all elements x of A.

Example The number 1 is an identity element for the operation of multiplication on the set $\mathbb{N}$ of natural numbers.

Example The number 0 is an identity element for the operation of addition on the set $\mathbb{Z}$ of integers.

Theorem 4.3 *A binary operation on a set cannot have more than one identity element.*

Proof Let e and f be identity elements for a binary operation $*$ on a set A. Then $e = e*f = f$. Thus there cannot be more than one identity element. ∎

4.7 Monoids

Definition A *monoid* consists of a set on which is defined an associative binary operation with an identity element.

We see immediately from the above definition that a semigroup is a monoid if and only if it has an identity element.

Definition A monoid $(A, *)$ is said to be *commutative* (or *Abelian*) if the binary operation $*$ is commutative.

Example The set $\mathbb{N}$ of natural numbers with the operation of multiplication is a commutative monoid. Indeed the operation of multiplication is both commutative and associative, and the identity element is the natural number 1.

Example The set $\mathbb{N}$ of natural numbers with the operation of addition is not a monoid, since there is no identity element for the operation of addition that belongs to the set of natural numbers.

Let a be an element of a monoid $(A, *)$. We define $a^0 = e$, where e is the identity element.

Theorem 4.4 *Let $(A, *)$ be a monoid, and let a be an element of A. Then $a^m * a^n = a^{m+n}$ for all non-negative integers m and n.*

Proof Any monoid is a semigroup. It therefore follows from Theorem 4.1 that $a^m * a^n = a^{m+n}$ when $m > 0$ and $n > 0$. It also follows directly from the definition of the identity element that the result is also true if $m = 0$ or if $n = 0$. ∎

Theorem 4.5 *Let $(A, *)$ be a monoid, and let a be an element of A. Then $(a^m)^n = a^{mn}$ for all non-negative integers m and n.*

Proof It follows directly from Theorem 4.2 that $(a^m)^n = a^{mn}$ whenever m and n are both positive. But this identity holds also when m or n is zero, since both sides of the identity are then equal to the identity element of the monoid. ∎

4.8 Inverses

Definition Let $(A, *)$ be a monoid with identity element e, and let x be an element of A. An element y of A is said to be the *inverse* of x if $x * y = y * x = e$. An element x of A is said to be *invertible* if there exists an element of A which is an inverse of x.

Theorem 4.6 *An element of a monoid can have at most one inverse.*

Proof Let $(A, *)$ be a monoid with identity element e, and let x, y and z be elements of A. Suppose that $x * y = y * x = e$ and $x * z = z * x = e$. Then

$$y = y * e = y * (x * z) = (y * x) * z = e * z = z,$$

and thus $y = z$. Thus an element of a monoid cannot have more than one inverse. ∎

Remark The above proof shows in fact that if x is an element of a monoid $(A, *)$, and if y and z are elements of A satisfying $y * x = x * z = e$, where e is the identity element of the monoid, then $y = z$.

Let $(A, *)$ be a monoid, and let x be an invertible element of A. We shall denote the inverse of x by x^{-1}. (This inverse element x^{-1} is uniquely determined by x, by Theorem 4.6.)

Theorem 4.7 *Let $(A, *)$ be a monoid, and let x and y be invertible elements of A. Then $x * y$ is also invertible, and $(x * y)^{-1} = y^{-1} * x^{-1}$.*

Proof Let e denote the identity element of the monoid. Then $x * x^{-1} = x^{-1} * x = e$ and $y * y^{-1} = y^{-1} * y = e$, and therefore

$$\begin{aligned}(x * y) * (y^{-1} * x^{-1}) &= ((x * y) * y^{-1}) * x^{-1} = (x * (y * y^{-1})) * x^{-1} \\ &= (x * e) * x^{-1} = x * x^{-1} = e, \\ (y^{-1} * x^{-1}) * (x * y) &= y^{-1} * (x^{-1} * (x * y)) = y^{-1} * ((x^{-1} * x) * y) \\ &= y^{-1} * (e * y) = y^{-1} * y = e.\end{aligned}$$

and thus the element $y^{-1} * x^{-1}$ has the properties required of an inverse of the element $x * y$. We conclude that $x * y$ is indeed invertible, and $(x * y)^{-1} = y^{-1} * x^{-1}$. ∎

Theorem 4.8 *Let $(A, *)$ be a monoid, let a and b be elements of A, and let x be an invertible element of A. Then $a = b * x$ if and only if $b = a * x^{-1}$. Similarly $a = x * b$ if and only if $b = x^{-1} * a$.*

Proof Let e denote the identity element of the monoid. Suppose that $a = b * x$. Then

$$a * x^{-1} = (b * x) * x^{-1} = b * (x * x^{-1}) = b * e = b.$$

Conversely, if $b = a * x^{-1}$, then

$$b * x = (a * x^{-1}) * x = a * (x^{-1} * x) = a * e = a.$$

Similarly if $a = x * b$ then

$$x^{-1} * a = x^{-1} * (x * b) = (x^{-1} * x) * b = e * b = b,$$

and, conversely, if $b = x^{-1} * a$ then

$$x * b = x * (x^{-1} * a) = (x * x^{-1}) * a = e * a = a. \quad ∎$$

Let $(A, *)$ be a monoid, and let a be an invertible element of A. We extend the definition of a^n to negative integers n by defining a^n to be the inverse $(a^q)^{-1}$ of a^q whenever $q > 0$ and $n = -q$.

Theorem 4.9 *Let $(A, *)$ be a monoid, and let a be an invertible element of A. Then $a^m * a^n = a^{m+n}$ for all integers m and n.*

Proof The proof breaks down into a case-by-case analysis, depending on the signs of the integers m and n.

The appropriate definitions ensure that the identity $a^m * a^n = a^{m+n}$ holds if $m = 0$ or if $n = 0$.

The result has already been verified if both m and n are positive (see Theorem 4.1 and Theorem 4.4).

Suppose that m and n are both negative. Then $a^m = (a^{-m})^{-1}$, $a^n = (a^{-n})^{-1}$ and $a^{m+n} = (a^{-(m+n)})^{-1}$. Now $a^{-n} * a^{-m} = a^{-n-m} = a^{-(m+n)}$. It follows from Theorem 4.7 that

$$a^{m+n} = (a^{-(m+n)})^{-1} = (a^{-n} * a^{-m})^{-1} = (a^{-m})^{-1} * (a^{-n})^{-1} = a^m * a^n.$$

The only remaining cases to consider are those when m and n have different signs.

Let p and q be non-negative integers. Now $a^{p+q} = a^p * a^q = a^q * a^p$. It follows from Theorem 4.8 that

$$a^p = a^{p+q} * a^{-q} = a^{-q} * a^{p+q}, \quad a^q = a^{p+q} * a^{-p} = a^{-p} * a^{p+q},$$

and hence

$$a^{-p} = a^q * a^{-(p+q)} = a^{-(p+q)} * a^q, \quad a^{-q} = a^p * a^{-(p+q)} = a^{-(p+q)} * a^p.$$

Suppose that $m < 0$, $n > 0$ and $m + n \geq 0$. On setting $p = -m$ and $q = m + n$ we see that $a^{m+n} = a^q = a^{-p} * a^{p+q} = a^m * a^n$. Next suppose that $m < 0$, $n > 0$ and $m + n < 0$. On setting $p = -m - n$ and $q = n$ we see that $a^{m+n} = a^{-p} = a^{-(p+q)} * a^q = a^m * a^n$. Next suppose that $m > 0$, $n < 0$ and $m + n \geq 0$. On setting $p = m + n$ and $q = -n$ we see that $a^{m+n} = a^p = a^{p+q} * a^{-q} = a^m * a^n$. Finally suppose that $m > 0$, $n < 0$ and $m + n < 0$. On setting $p = m$ and $q = -m - n$ we see that $a^{m+n} = a^{-q} = a^p * a^{-(p+q)} = a^m * a^n$. The result has now been verified for all integers m and n, as required. ∎

Theorem 4.10 *Let $(A, *)$ be a monoid, and let a be an invertible element of A. Then $(a^m)^n = a^{mn}$ for all integers m and n.*

Proof Let m be an integer. First we prove by induction on n that $(a^m)^n = a^{mn}$ for all positive integers n. The result clearly holds when $n = 1$. Suppose $(a^m)^s = a^{ms}$ for some positive integer s. It then follows from Theorem 4.9 that

$$(a^m)^{s+1} = (a^m)^s * a^m = a^{ms} * a^m = a^{m(s+1)}.$$

It follows from the Principle of Mathematical Induction that $(a^m)^n = a^{mn}$ for all positive integers n. The result is also true when $n = 0$, since both sides of the identity are then equal to the identity element of the monoid. Finally suppose that n is a negative integer. Then $n = -q$ for some positive integer q, and $(a^m)^q = a^{mq}$. On taking the inverses of both sides of this identity, we find that

$$(a^m)^n = ((a^m)^q)^{-1} = (a^{mq})^{-1} = a^{-mq} = a^{mn},$$

as required. We can now conclude that the identity $(a^m)^n = a^{mn}$ holds for all integers m and n. ∎

4.9 Groups

Definition A *group* consists of a set A together with a binary operation $*$ on A with the following properties:—

(i) $x * (y * z) = (x * y) * z$ for all elements x, y and z of A (i.e., the operation $*$ is associative);

(ii) there exists an element e of A with the property that $e * x = x * e = x$ for all elements x of A (i.e., there exists an identity element e for the binary operation $*$ on A);

(iii) given any element x of A, there exists an element y of A satisfying $x * y = y * x = e$ (i.e., every element of A is invertible).

We see immediately from this definition that a *group* can be characterized as a monoid in which every element is invertible.

Definition A group $(A, *)$ is said to be *commutative* (or *Abelian*) if the binary operation $*$ is commutative.

Example The set of integers with the operation of addition is a commutative group.

Example The set of real numbers with the operation of addition is a commutative group.

Example The set of non-zero real numbers with the operation of multiplication is a commutative group.

Example The set of integers with the operation of multiplication is not a group, since not every element is invertible. Indeed the only integers that are invertible are $+1$ and -1.

Example Let n be a natural number, and let

$$\mathbb{Z}_n = \{0, 1, \ldots, n-1\}.$$

Any integer k may be expressed uniquely in the form $k = qn + r$ for some integers q and r with $0 \leq r < n$. (When k is positive, q and r are the quotient and remainder respectively, when k is divided by n in integer arithmetic.) Then r is the unique element of $\mathbb{Z}_n$ for which $k - r$ is divisible by n. In particular, given any elements x and y of $\mathbb{Z}_n$, there exist unique elements s and p of $\mathbb{Z}_n$ such that $x + y - s$ and $xy - p$ are divisible by n. We define $x \oplus_n y = s$ and $x \otimes_n y = p$. Then $\oplus_n$ and $\otimes_n$ are binary operations on the set $\mathbb{Z}_n$.

We show that the binary operation $\oplus_n$ is associative. Let x, y and z be integers belonging to $\mathbb{Z}_n$, and let $u = x \oplus_n y$ and $v = y \oplus_n z$. Then $x + y - u$ and $y + z - v$ are both divisible by n. Now

$$(u + z) - (x + v) = (y + z - v) - (x + y - u).$$

It follows that $(u + z) - (x + v)$ is divisible by n, and hence $u \oplus_n z = x \oplus_n v$. Thus $(x \oplus_n y) \oplus_n z = x \oplus_n (y \oplus_n z)$.

We also show that the binary operation $\otimes_n$ is associative. Let x, y and z be integers belonging to $\mathbb{Z}_n$, and let $p = x \otimes_n y$ and $q = y \otimes_n z$. Then $xy - p$ and $yz - q$ are both divisible by n. Now

$$pz - xq = x(yz - q) - (xy - p)z.$$

It follows that $pz - xq$ is divisible by n, and hence $p \otimes_n z = x \otimes_n q$. Thus $(x \otimes_n y) \otimes_n z = x \otimes_n (y \otimes_n z)$.

Now $0 \oplus_n x = x \oplus_n 0 = x$ and $1 \otimes_n x = x \otimes_n 1 = x$ for all $x \in \mathbb{Z}_n$. It follows that $(\mathbb{Z}_n, \oplus_n)$ is a monoid with identity element 0, and $(\mathbb{Z}_n, \otimes_n)$ is a monoid with identity element 1.

Every element x of the monoid $(\mathbb{Z}_n, \oplus_n)$ is invertible: the inverse of x is $n - x$ if $x \neq 0$, and is 0 if $x = 0$. Thus $(\mathbb{Z}_n, \oplus_n)$ is a group.

However $(\mathbb{Z}_n, \otimes_n)$ is not a group if $n > 1$. Indeed 0 is not an invertible element, since $0 \otimes_n x = 0$ for all elements x of $\mathbb{Z}_n$, and therefore there cannot exist any element x of $\mathbb{Z}_n$ for which $0 \otimes_n x = 1$.

It can be shown that an element x of $(\mathbb{Z}_n, \otimes_n)$ is invertible in this monoid if and only if the highest common factor of x and n is equal to 1. It follows from this that the non-zero elements of $\mathbb{Z}_n$ constitute a group under $\otimes_n$ if and only if the natural number n is a prime number.

Let us consider the particular case when $n = 9$. The 'multiplication table' for the monoid $(\mathbb{Z}_9, \otimes_9)$ is the following:—

$\otimes_9$	0	1	2	3	4	5	6	7	8
0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8
2	0	2	4	6	8	1	3	5	7
3	0	3	6	0	3	6	0	3	6
4	0	4	8	3	7	2	6	1	5
5	0	5	1	6	2	7	3	8	4
6	0	6	3	0	6	3	0	6	3
7	0	7	5	3	1	8	6	4	2
8	0	8	7	6	5	4	3	2	1

From this table we see that the invertible elements are 1, 2, 4, 5, 7 and 8. Indeed $1 \otimes_9 1 = 1$, $2 \otimes_9 5 = 1$, $4 \otimes_9 7 = 1$, $8 \otimes_9 8 = 1$.

4.10 Homomorphisms and Isomorphisms

Definition Let $(A, *)$ and $(B, *)$ be semigroups, monoids or groups. A function $f: A \to B$ from A to B is said to be a *homomorphism* if $f(x * y) = f(x) * f(y)$ for all elements x and y of A.

Example Let q be an integer, and let $f: \mathbb{Z} \to \mathbb{Z}$ be a the function from the set of integers to itself defined by $f(n) = qn$ for all integers n. Then f is a homomorphism from the group $(\mathbb{Z}, +)$ to itself, since

$$f(m + n) = q(m + n) = qm + qn = f(m) + f(n)$$

for all integers m and n.

Example Let $\mathbb{R}^*$ denote the set of non-zero real numbers, let a be a non-zero real number, and let $f: \mathbb{Z} \to \mathbb{R}^*$ be the function defined by $f(n) = a^n$ for all integers m and n. Then $f: \mathbb{Z} \to \mathbb{R}^*$ is a homomorphism from the group $(\mathbb{Z}, +)$ of integers under addition to the group $(\mathbb{R}^*, \times)$ of non-zero real numbers under multiplication, since

$$f(m + n) = a^{m+n} = a^m a^n = f(m)f(n)$$

for all integers m and n.

Example This last example can be generalized. Let a be an invertible element of a monoid $(A, *)$, and let $f: \mathbb{Z} \to A$ be the function from $\mathbb{Z}$ to A defined by $f(n) = a^n$. Then this function is a homomorphism from the group $(\mathbb{Z}, +)$ of integers under addition to the monoid $(A, *)$ since it follows from Theorem 4.9 that

$$f(m+n) = a^{m+n} = a^m * a^n = f(m) * f(n)$$

for all integers m and n.

We recall that a function $f: A \to B$ is said to be *injective* if distinct elements of A get mapped to distinct elements of B (i.e., if x and y are elements of A and if $x \neq y$ then $f(x) \neq f(y)$). Also a function $f: A \to B$ is said to be *surjective* if each element of B is the image $f(a)$ of at least one element a of A. A function $f: A \to B$ is said to be *bijective* if it is both injective and surjective. One can prove that a function $f: A \to B$ has a well-defined inverse $f^{-1}: B \to A$ if and only if it is bijective.

Definition Let $(A, *)$ and $(B, *)$ be semigroups, monoids or groups. A function $f: A \to B$ from A to B is said to be an *isomorphism* if it is both a homomorphism and a bijective function.

Theorem 4.11 *Let $(A, *)$ and $(B, *)$ be semigroups, monoids or groups. Then the inverse $f^{-1}: B \to A$ of any isomorphism $f: A \to B$ is itself an isomorphism.*

Proof The inverse $f^{-1}: B \to A$ of an isomorphism $f: A \to B$ is itself a bijective function whose inverse is the function $f: A \to B$. It remains to show that $f^{-1}: B \to A$ is a homomorphism. Let u and v be elements of B, and let $x = f^{-1}(u)$ and $y = f^{-1}(v)$. Then $u = f(x)$ and $v = f(y)$, and therefore

$$f(x * y) = f(x) * f(y) = u * v$$

and therefore

$$f^{-1}(u * v) = x * y = f^{-1}(u) * f^{-1}(v),$$

showing that the function $f^{-1}: B \to A$ is a homomorphism from $(B, *)$ to $(A, *)$, as required. ∎

Definition Let $(A, *)$ and $(B, *)$ be semigroups, monoids or groups. If there exists an isomorphism from $(A, *)$ to $(B, *)$ then $(A, *)$ and $(B, *)$ are said to be *isomorphic*.

5 Formal Languages

5.1 Alphabets and Words

Let A be a finite set. We shall refer to this set A as an *alphabet* and we may refer to the elements of A as *letters*. (For example, the set A might be the set of letters in the English language, or in any other world language, or the set $\{0, 1\}$ of binary digits, or the set $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ of decimal digits.)

For any natural number n, we define a *word* of length n over the alphabet A to be a string of the form $a_1 a_2 \dots a_n$ in which $a_i \in A$ for $i = 1, 2, \dots, n$. We shall denote by A^n the set of all words of length n over the alphabet A. In particular, $A^1 = A$.

Example Let $A = \{a, b, c\}$. Then

$$\begin{aligned} A^2 &= \{aa, ab, ac, ba, bb, bc, ca, cb, cc\} \\ A^3 &= \{aaa, aab, aac, aba, abb, abc, \dots, ccc\} \end{aligned}$$

Remark The set A^n can be identified with the Cartesian product $A \times A \times \cdots \times A$ of n copies of the set A. The elements of this Cartesian product are ordered n-tuples $(a_1, a_2, \dots, a_n)$ whose components $a_1, a_2, \dots, a_n$ are elements of the set A. However, in the interests of brevity, it is convenient to drop the parentheses and commas, denoting any element $(a_1, a_2, \dots, a_n)$ of this Cartesian product by the corresponding string or word $a_1 a_2 \dots a_n$ of length n.

We denote by A^+ the union of the sets A^n for all natural numbers n, so that

$$A^+ = \bigcup_{n=1}^{\infty} A^i = A \cup A^2 \cup A^3 \cup A^4 \cup \cdots.$$

The set A^+ is thus the set of all words of positive length over the alphabet n.

Example Let $A = \{0, 1\}$. Then A^+ is the set of all binary strings whose length is finite and non-zero, and contains strings such as 1, 0, 10, 101, 010, 010101, and 000010010.

We introduce also an *empty word* ε, which we regard as a word of length zero. This may be thought of as the empty string, not involving any of the letters from the alphabet A. We define $A^0 = \{\varepsilon\}$, and we denote by A^* the set $\{\varepsilon\} \cup A^+$ obtained by adjoining the empty word ε to the set A^+ of words of positive length over the alphabet A. Thus

$$A^* = \{\varepsilon\} \cup A^+ = \bigcup_{n=0}^{\infty} A^i = \{\varepsilon\} \cup A \cup A^2 \cup A^3 \cup A^4 \cup \cdots.$$

Each word in A^* has a length which is a non-negative integer. The empty word is the only word in A^* whose length is zero.

We denote the length of a word w by $|w|$.

Definition Let A be a finite set, and let w_1 and w_2 be words over the alphabet A, with $w_1 = a_1a_2 \dots a_m$ and $w_2 = b_1b_2 \dots b_n$. The *concatenation* of the words w_1 and w_2 is the word $w_1 \circ w_2$, where

$$w_1 \circ w_2 = a_1a_2 \dots a_mb_1b_2 \dots b_n.$$

The concatenation $w_1 \circ w_2$ of two words w_1 and w_2 may also be denoted by w_1w_2.

Example Let A be the set of lower case letters in the English alphabet, and let w_1 and w_2 be the words 'book' and 'case' respectively. Then $w_1 \circ w_2$ is the word 'bookcase' and $w_2 \circ w_1$ is the word 'casebook'. Note that, in this instance, $w_1 \circ w_2 \neq w_2 \circ w_1$.

Note that $|w_1 \circ w_2| = |w_1| + |w_2|$ for all words w_1 and w_2 over some alphabet.

The operation $\circ$ of concatenation on the set of words over some alphabet is not commutative if that alphabet has more than one element. Indeed if a and b are distinct elements of this alphabet then $a \circ b$ is the string ab, and $b \circ a$ is the string ba, and therefore $a \circ b \neq b \circ a$.

Let w_1, w_2 and w_3 be words over some alphabet A. Then $(w_1 \circ w_2) \circ w_3 = w_1 \circ (w_2 \circ w_3)$. Indeed if

$$w_1 = a_1a_2 \dots a_m, \quad w_2 = b_1b_2 \dots b_n, \quad w_3 = c_1c_2 \dots c_p,$$

then $(w_1 \circ w_2) \circ w_3$ and $w_1 \circ (w_2 \circ w_3)$ both denote the word

$$a_1a_2 \dots a_mb_1b_2 \dots b_nc_1c_2 \dots c_p.$$

The empty word ε has the property that $\varepsilon \circ w = w \circ \varepsilon = w$ for all words w over an alphabet A.

The following theorem follows directly from these observations.

Theorem 5.1 *Let A be a finite set. Then $(A^*, \circ)$ is a monoid, where the set A^* is the set of words over the alphabet A, and $\circ$ is the operation of concatenation of words. The identity element of this monoid is the empty word ε.*

Definition Let A be a finite set. A *language* over A is a subset of A^*. A language L over A is said to be a *formal language* if there is some finite set of rules or algorithm that will generate all the words that belong to L and no others.

Let A be a finite set. The union and intersection of any finite collection of languages over A are themselves languages over A. In particular, the union $L_1 \cup L_2$ and intersection $L_1 \cap L_2$ of two languages L_1 and L_2 over A is a language over A.

The *concatenation* of languages L_1 and L_2 over A is the language $L_1 \circ L_2$, where

$$L_1 \circ L_2 = \{w_1 \circ w_2 \in A^* : w_1 \in L_1 \text{ and } w_2 \in L_2\}.$$

The concatenation $L_1 \circ L_2$ of the languages L_1 and L_2 may also be denoted by $L_1 L_2$.

Given any language L, we can form languages L^n for all natural numbers n, where $L^1 = L$ and $L^n = L \circ L^{n-1}$ for all natural numbers n. The associativity of the concatenation operation ensures that $L^m \circ L^n = L^{m+n}$ for all natural numbers m and n. We define $L^0 = \{\epsilon\}$. The language L then determines languages L^+ and L^* over A, where

$$L^+ = \bigcup_{n=1}^{\infty} L^i = L \cup L^2 \cup L^3 \cup L^4 \cup \cdots$$

and

$$L^* = \{\varepsilon\} \cup L^+ = \bigcup_{n=0}^{\infty} L^i = \{\varepsilon\} \cup L \cup L^2 \cup L^3 \cup L^4 \cup \cdots.$$

5.2 Simple Grammars to Generate English Sentences

We describe a simple grammar to construct a certain collection of English sentences. This grammar is specified by the following collection of *productions* or replacement rules:

$$\begin{aligned}
\langle \text{S} \rangle &\rightarrow \langle \text{NP} \rangle \langle \text{VP} \rangle \\
\langle \text{VP} \rangle &\rightarrow \langle \text{V} \rangle \langle \text{NP} \rangle \\
\langle \text{NP} \rangle &\rightarrow \langle \text{T} \rangle \langle \text{QN} \rangle \\
\langle \text{QN} \rangle &\rightarrow \langle \text{Adj} \rangle \langle \text{QN} \rangle \\
\langle \text{QN} \rangle &\rightarrow \langle \text{N} \rangle \\
\langle \text{T} \rangle &\rightarrow \text{the} \\
\langle \text{N} \rangle &\rightarrow \text{cat}
\end{aligned}$$

$$
\begin{aligned}
\langle \mathrm{N} \rangle &\rightarrow \text{dog} \\
\langle \mathrm{Adj} \rangle &\rightarrow \text{black} \\
\langle \mathrm{Adj} \rangle &\rightarrow \text{old} \\
\langle \mathrm{Adj} \rangle &\rightarrow \text{small} \\
\langle \mathrm{V} \rangle &\rightarrow \text{saw} \\
\langle \mathrm{V} \rangle &\rightarrow \text{chased}
\end{aligned}
$$

Here $\langle \mathrm{S} \rangle$, $\langle \mathrm{VP} \rangle$, $\langle \mathrm{NP} \rangle$, $\langle \mathrm{QN} \rangle$, $\langle \mathrm{T} \rangle$, $\langle \mathrm{N} \rangle$, $\langle \mathrm{Adj} \rangle$, $\langle \mathrm{V} \rangle$, represent certain phrases or words that may form part of sentences generated by the grammar. They may be interpreted as follows:—

- $\langle \mathrm{S} \rangle$ represents the sentence to be generated by the grammar;
- $\langle \mathrm{VP} \rangle$ represents a 'verb phrase';
- $\langle \mathrm{NP} \rangle$ represents a 'noun phrase';
- $\langle \mathrm{QN} \rangle$ represents a noun optionally preceded by one or more adjectives;
- $\langle \mathrm{T} \rangle$ represents the definite article 'the' (which will subsequently replace it);
- $\langle \mathrm{N} \rangle$ represents a noun chosen from the set the set $\{\text{dog}, \text{cat}\}$;
- $\langle \mathrm{Adj} \rangle$ represents an adjective chosen from the set $\{\text{black}, \text{old}, \text{small}\}$;
- $\langle \mathrm{V} \rangle$ represents a verb chosen from the set $\{\text{saw}, \text{chased}\}$;

Each of these productions specifies that the entity on the left hand side of the arrow may be replaced by the string on the right hand side of the arrow.

We may apply these productions successively, starting with the symbol $\langle \mathrm{S} \rangle$, in order to obtain the sentences generated by the grammar.

Example We may generate the sentence 'The dog chased the old black cat'

as follows:

⟨S⟩	⇒	⟨NP⟩⟨VP⟩	(⟨S⟩	→	⟨NP⟩⟨VP⟩)
	⇒	⟨T⟩⟨QN⟩⟨VP⟩	(⟨NP⟩	→	⟨T⟩⟨QN⟩)
	⇒	the ⟨QN⟩⟨VP⟩	(⟨T⟩	→	the)
	⇒	the ⟨N⟩⟨VP⟩	(⟨QN⟩	→	⟨N⟩)
	⇒	the dog ⟨VP⟩	(⟨N⟩	→	dog)
	⇒	the dog ⟨V⟩⟨NP⟩	(⟨VP⟩	→	⟨V⟩⟨NP⟩)
	⇒	the dog chased ⟨NP⟩	(⟨V⟩	→	chased)
	⇒	the dog chased ⟨T⟩⟨QN⟩	(⟨NP⟩	→	⟨T⟩⟨QN⟩)
	⇒	the dog chased the ⟨QN⟩	(⟨T⟩	→	the)
	⇒	the dog chased the ⟨Adj⟩⟨QN⟩	(⟨QN⟩	→	⟨Adj⟩⟨QN⟩)
	⇒	the dog chased the old ⟨QN⟩	(⟨Adj⟩	→	old)
	⇒	the dog chased the old ⟨Adj⟩⟨QN⟩	(⟨QN⟩	→	⟨Adj⟩⟨QN⟩)
	⇒	the dog chased the old black ⟨QN⟩	(⟨Adj⟩	→	black)
	⇒	the dog chased the old black ⟨N⟩	(⟨QN⟩	→	⟨N⟩)
	⇒	the dog chased the old black cat.	(⟨N⟩	→	cat)

The production used at each step of the derivation is specified on the right. At each stage a single instance of ⟨S⟩, ⟨VP⟩, ⟨NP⟩, ⟨QN⟩, ⟨T⟩, ⟨N⟩, ⟨Adj⟩ or ⟨V⟩ is replaced by the appropriate string.

In grammars such as the one we are studying, words such as 'the', 'black', 'old', 'small', 'cat', 'dog', 'chased', 'saw' are referred to as *terminals*. Entities such as ⟨S⟩, ⟨VP⟩, ⟨NP⟩, ⟨QN⟩, ⟨T⟩, ⟨N⟩, ⟨Adj⟩ and ⟨V⟩ are referred to as *nonterminals*. Each production in a *context-free grammar* specifies that a single occurrence of the nonterminal specified to the left of the arrow may be replaced by the string specified to the right of the arrow. Such replacements are applied, one at a time, to transform strings made up of terminals and nonterminals to other strings of the same kind.

Grammars defined by means of productions may be specified in a more compact form, introduced by John Backus and Peter Naur, and employed in a report on the high-level programming language ALGOL-60 published in 1960. Our grammar for simple English sentences may be presented in Backus-Naur form as follows:

$$
\begin{array}{rcl}
\langle \mathrm{S} \rangle & \rightarrow & \langle \mathrm{NP} \rangle \langle \mathrm{VP} \rangle \\
\langle \mathrm{VP} \rangle & \rightarrow & \langle \mathrm{V} \rangle \langle \mathrm{NP} \rangle \\
\langle \mathrm{NP} \rangle & \rightarrow & \langle \mathrm{T} \rangle \langle \mathrm{QN} \rangle \\
\langle \mathrm{QN} \rangle & \rightarrow & \langle \mathrm{Adj} \rangle \langle \mathrm{QN} \rangle \mid \langle \mathrm{N} \rangle \\
\langle \mathrm{T} \rangle & \rightarrow & \mathrm{the}
\end{array}
$$

$$
\begin{aligned}
\langle \mathrm{N} \rangle &\rightarrow \text{cat} \mid \text{dog} \\
\langle \mathrm{Adj} \rangle &\rightarrow \text{black} \mid \text{old} \mid \text{small} \\
\langle \mathrm{V} \rangle &\rightarrow \text{saw} \mid \text{chased}
\end{aligned}
$$

Each item in this list specifies that the nonterminal occurring to the left of the arrow may be replaced by one of various alternatives; the alternatives are separated by the meta-character |.

One may verify that the sentences generated by the grammar are of the following form: the sentence consists of the definite article 'the', optionally followed by one or more of the adjectives 'black', 'old' and 'small', followed by one of the nouns 'cat' and 'dog', followed by one of the verbs 'saw' and 'chased', followed by the definite article 'the', optionally followed by one or more of the listed adjectives, followed by one of the listed nouns. From this description one may verify that the sentences generated by simple grammar we have described are the same as those generated by the grammar specified in Backus-Naur form as follows:—

$$
\begin{aligned}
\langle \mathrm{S} \rangle &\rightarrow \text{the } \langle \mathrm{T1} \rangle \\
\langle \mathrm{T1} \rangle &\rightarrow \text{black } \langle \mathrm{T1} \rangle \mid \text{old } \langle \mathrm{T1} \rangle \mid \text{small } \langle \mathrm{T1} \rangle \mid \text{cat } \langle \mathrm{T2} \rangle \mid \text{dog } \langle \mathrm{T2} \rangle \\
\langle \mathrm{T2} \rangle &\rightarrow \text{saw } \langle \mathrm{T3} \rangle \mid \text{chased } \langle \mathrm{T3} \rangle \\
\langle \mathrm{T3} \rangle &\rightarrow \text{the } \langle \mathrm{T4} \rangle \\
\langle \mathrm{T4} \rangle &\rightarrow \text{black } \langle \mathrm{T4} \rangle \mid \text{old } \langle \mathrm{T4} \rangle \mid \text{small } \langle \mathrm{T4} \rangle \mid \text{cat } \langle \mathrm{F} \rangle \mid \text{dog } \langle \mathrm{F} \rangle \\
\langle \mathrm{F} \rangle &\rightarrow \varepsilon
\end{aligned}
$$

This grammar is an example of a *regular grammar*. A regular grammar is determined by productions in which a single nonterminal is replaced, either by a terminal followed by a nonterminal, or by a single terminal, or by the empty string ε.

Example The sentence 'The dog chased the old black cat' is generated in the regular grammar presented above as follows:

$$
\begin{aligned}
\langle \mathrm{S} \rangle &\Rightarrow \text{the } \langle \mathrm{T1} \rangle \\
&\Rightarrow \text{the dog } \langle \mathrm{T2} \rangle \\
&\Rightarrow \text{the dog chased } \langle \mathrm{T3} \rangle \\
&\Rightarrow \text{the dog chased the } \langle \mathrm{T4} \rangle \\
&\Rightarrow \text{the dog chased the old } \langle \mathrm{T4} \rangle \\
&\Rightarrow \text{the dog chased the old black } \langle \mathrm{T4} \rangle \\
&\Rightarrow \text{the dog chased the old black cat } \langle \mathrm{F} \rangle \\
&\Rightarrow \text{the dog chased the old black cat.}
\end{aligned}
$$

If a language is generated by a regular grammar then it is possible to construct a *finite state acceptor* for that language. This is a *finite state machine* which may be used to determine whether or not a given string belongs to the language.

We describe a finite state acceptor for the collection of sentences generated by the regular grammar described above. This machine has a finite number of internal states. One of these states is the *initial state* of the machine. Some of the states of a finite state acceptor are regarded as *final states*. Words taken from the list

the, black, old, small, cat, dog, saw, chased

are successively input into the machine. Each time one of these words is input the machine either remains in the state it is currently in, or it makes a transition to some other internal state, determined by the current state and the input word. We consider such a machine with seven internal states, which we label as S, T1, T2, T3, T4, F and E. The following table describes the effect of inputting each word:

	the	black	old	small	cat	dog	saw	chased
S	T1	E	E	E	E	E	E	E
T1	E	T1	T1	T1	T2	T2	E	E
T2	E	E	E	E	E	E	T3	T3
T3	T4	E	E	E	E	E	E	E
T4	E	T4	T4	T4	F	F	E	E
F	E	E	E	E	E	E	E	E
E	E	E	E	E	E	E	E	E

The words label the columns of the table, the internal states label the rows, and each entry in the table specifies the state that results when the current state is that labelling the row and the input word is that labelling the column. For example, if the machine is in state T1, and if the input word is 'cat', then the internal state of the machine is changed to the state T2. The state E may be regarded as an 'error state': the machine enters this state whenever a word is input that may not occur at the relevant position in the sentence. Moreover, once the machine is in the state E, it remains in this state, no matter which word is input. The state S is the initial or starting state. There is a single final state, which is the state F. A finite sequence of words is accepted by the machine if and only if successively inputting these words causes the machine to move from state S to state F.

5.3 Well-Formed Formulae in Logic

We shall investigate the grammar of well-formed formulae in the Propositional Calculus. We begin with a discussion of basic principles of this calculus.

Let p and q be *Boolean variables*. Such variables may represent propositions that might be true in certain circumstances, or false in other circumstances. A Boolean variable may therefore take on one of two values: true (**T**) or false (**F**).

Example Let the Boolean variable p represent the proposition $(x > y)$, where x and y are arbitrary real numbers. If $x = 10$ and $y = 5$ then p is true. But if $x = 5$ and $y = 10$ then p is false.

In the Propositional Calculus we also have two Boolean constants. We shall denote one of these by **T**: it may represent a proposition that is true in all circumstances. We shall denote the other constant by **F**: it may represent a proposition that is false in all circumstances.

In the Propositional Calculus we may build more complicated formulae out of simpler formulae using the operations of *negation* $\neg$, *conjunction* $\wedge$ and and *disjunction* $\vee$. Negation is a unary operation, whereas conjunction and disjunction are binary operations.

Let p be a Boolean variable. The *negation* $\neg p$ of p has the property that it is true whenever p is false, and it is false whenever p is true. The relationship between p and $\neg p$ is therefore expressed by the following truth table:—

p	$\neg p$
T	**F**
F	**T**

Let p and q be Boolean variables. The *conjunction* $p \wedge q$ of p and q is true if and only if both p and q are true. The *disjunction* $p \vee q$ of p and q is true if and only if at least one of p and q is true. The relationship between p, q, $p \wedge q$ and $p \vee q$ are therefore expressed by the following truth tables:—

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

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

The conjunction $p \wedge q$ may be thought of as representing the proposition p AND q. Similarly the disjunction $p \vee q$ may be thought of as representing the proposition p OR q.

We may build up more complicated formulae using these basic operations of negation, conjunction and disjunction. For example, if p, q and r are Boolean variables, and if the Boolean variable s is used to represent the conjunction $p \wedge q$ of p and q, then we may write

$$\neg s = \neg(p \wedge q), \quad s \wedge r = (p \wedge q) \wedge r,$$

$$s \vee r = (p \wedge q) \vee r, \quad r \vee s = r \vee (p \wedge q), \quad \text{etc.}$$

Consider the propositions represented by the formulae $(p \wedge q) \wedge r$ and $p \wedge (q \wedge r)$. These two propositions are true if and only if each of the propositions p, q and r is true. They are false if any one of the propositions p, q and r is false. It follows that no ambiguity will result if we write $p \wedge q \wedge r$ in place of $(p \wedge q) \wedge r$ or $p \wedge (q \wedge r)$. More generally, we can define the conjunction of any finite number of propositions: the *conjunction* $p_1 \wedge p_2 \wedge \cdots \wedge p_n$ of propositions $p_1, p_2, \ldots, p_n$ is true if and only if every one of the propositions $p_1, p_2, \ldots, p_n$ is true. We may define in a similar fashion the disjunction of any finite number of propositions: the *disjunction* $p_1 \vee p_2 \vee \cdots \vee p_n$ of propositions $p_1, p_2, \ldots, p_n$ is true if and only if at least one of the propositions $p_1, p_2, \ldots, p_n$ is true.

Let us now consider what meaning, if any, one might assign to a formula such as '$p \wedge q \vee r$'. One might consider interpreting a formula of this form either as $(p \wedge q) \vee r$ or as $p \wedge (q \vee r)$. The following truth table exhibits the dependence of the truth values of these two latter formulae on those of p, q and r:—

p	q	r	$p \wedge q$	$q \vee r$	$(p \wedge q) \vee r$	$p \wedge (q \vee r)$
T	**T**	**T**	**T**	**T**	**T**	**T**
T	**T**	**F**	**T**	**T**	**T**	**T**
T	**F**	**T**	**F**	**T**	**T**	**T**
T	**F**	**F**	**F**	**F**	**F**	**F**
F	**T**	**T**	**F**	**T**	**T**	**F**
F	**T**	**F**	**F**	**T**	**F**	**F**
F	**F**	**T**	**F**	**T**	**T**	**F**
F	**F**	**F**	**F**	**F**	**F**	**F**

We see from this truth table that $(p \wedge q) \vee r$ and $p \wedge (q \vee r)$ do not represent equivalent propositions. For example, if p is false, q is true and r is true, then $(p \wedge q) \vee r$ is true, but $p \wedge (q \vee r)$ is false.

We would need to resolve this ambiguity in some way if we were to admit expressions such as '$p \wedge q \vee q$'. One approach would be to assign higher precedence to one or other of the binary operations $\wedge$ and $\vee$. (This would correspond to the convention in evaluating expressions in ordinary arithmetic

and algebra, where multiplication is assigned a higher precedence than addition.) A second possible approach would involve assigning equal precedence to the two operations $\wedge$ and $\vee$ whilst adopting the convention that evaluations of formulae in the Propositional Calculus involving these operations proceed from left to right in the absence of any parentheses indicating the order in which the operations are to be performed. (This would correspond to a standard convention in evaluating expressions in ordinary arithmetic and algebra involving additions and subtractions.)

However a sensible approach would involve regarding a formula such as '$p \wedge q \vee r$' as being ill-formed, on the grounds that it is inherently ambiguous in the absence of parentheses that would specify the order in which the operations $\wedge$ and $\vee$ are to be performed. One would not then seek to assign any truth value to such a formula, any more than one would seek to assign a truth value to a jumble of symbols such as '$p) \wedge \vee(((qr$'.

We are then led to the problem of providing a formal specification to determine which strings of characters involving 'p', 'q', 'r' etc., '$\neg$', '$\wedge$', '$\vee$' '(' and ')' are to be regarded as well-formed formulae in the Propositional Calculus. A related problem is that of designing an algorithm to determine whether or not a string involving these characters is to be regarded as a well-formed formula of the Propositional Calculus.

A formula of the Propositional Calculus consists of a string of characters taken from some finite set. This set would contain characters such as $\neg$, $\wedge$ and $\vee$ to denote the operations of negation, conjunction and disjunction respectively. It might also contain parentheses '(' and ')' that can be used to determine in the usual fashion the order in which the binary operations are to be performed and the subformulae to which they are to be applied. We could introduce characters **T** and **F** to denote the Boolean constants 'true' and 'false' respectively. It remains to consider how Boolean variables are to be represented. We could certainly use single letters p, q, r, s. But this would only enable us to write down formulae with at most four distinct propositional variables. Were we to use single letters from the English alphabet in both upper and lower case to denote propositional variables, this would restrict us to formulae with at most fifty-two distinct Boolean variables. But there should be no limit to the number of distinct Boolean variables that we could introduce into a well-formed formula. We therefore need a scheme for representing unlimited quantities of Boolean variables within our formula. We could do this by using p', p'', p''', p'''' etc., in addition to single letters such as p. Here p'''', for example, is to be regarded as a string of length 5, consisting of the letter p, followed by four instances of the prime character $'$. Accordingly we shall specify that a propositional variable is to be represented by the letters p, q, r, s, either alone or else followed by a string consisting

of any number of prime characters. (The choice of which letters to use is of course arbitrary; any similar choice would serve just as well.) The formulae of the Propositional Calculus may then be regarded as strings (or words) over the alphabet A, where

$$A = \{\neg, \wedge, \vee, (,), \mathbf{T}, \mathbf{F}, p, q, r, s, '\}.$$

The elements of A^* (i.e., the words over the alphabet A) are then strings of characters taken from the set A. Some of them, such as '$(p \wedge p') \vee (p'' \wedge p''')$', represent propositions whose truth values are determined unambiguously from the truth values of the Boolean variables occurring within them. Others, such as '$)p\neg) \wedge \vee \vee (\mathbf{T}p'''$' are pure gibberish. Our task is then to provide some sort of formal specification which determines which of the strings belonging to A^* are to be regarded as well-formed formulae of the Propositional Calculus. The collection of well-formed formulae is then a language over the alphabet A.

The method by which we specify the well-formed formulae is an example of a *context-free grammar*. For this, we introduce a set N of *nonterminals*. We represent each nonterminal by an appropriate identifier enclosed within angle brackets '$\langle$' and '$\rangle$'. For example we shall use the nonterminal '$\langle$wff$\rangle$' to represent an arbitrary well-formed formula. Other nonterminals shall be used to represent well-formed formulae that are of a special form. For example, the nonterminal '$\langle$negation$\rangle$' will represent a well-formed formula that is the negation of some other well-formed formula. Other nonterminals may represent things such as the letters p, q, r and s of the English alphabet.

We shall refer to elements of the set A as *terminals*. (In any context-free grammar, the *terminals* are the elements of the alphabet over which the language specified by that grammar is defined.)

The context-free grammar will then consist of a finite collection of *productions*. Each production specifies that a certain nonterminal may be replaced by some string whose elements are terminals or nonterminals. A word in A^* is then said to be *generated* by the grammar if some succession of replacements determined by productions in the grammar transforms the nonterminal $\langle$wff$\rangle$ into the given word.

Our grammar will include three productions in which the nonterminal $\langle$wff$\rangle$ occurs on the left hand side. These are

$$\begin{aligned} \langle\text{wff}\rangle &\rightarrow (\langle\text{wff}\rangle) \\ \langle\text{wff}\rangle &\rightarrow \langle\text{atom}\rangle \\ \langle\text{wff}\rangle &\rightarrow \langle\text{compound}\rangle \end{aligned}$$

The effect of including the first of these productions is to ensure that, whenever a well-formed formula is enclosed within parentheses, the resulting for-

mula is also well-formed. For example, the formulae $(p \wedge q)$, $((p \wedge q))$, $(((p \wedge q)))$, etc., are obtained in this way from the well-formed formula $p \wedge q$, and our grammar will therefore ensure that these formulae are also well-formed.

If a well-formed formula is not merely other well-formed formulae enclosed within parentheses, then it may be regarded as being either an atomic formula or a compound formula. The atomic formulae are the Boolean constants **T** and **F** and variables such as p, q, r, s, p', p'' etc. The compound formulae are those that are constructed out of shorter well-formed formulae using the operations of negation, conjunction and disjunction. The production

$$\langle \text{wff} \rangle \to \langle \text{atom} \rangle$$

allows us to replace the nonterminal $\langle \text{wff} \rangle$ by $\langle \text{atom} \rangle$ when the well-formed formula we are seeking to generate is a Boolean constant or a Boolean variable. The production

$$\langle \text{wff} \rangle \to \langle \text{compound} \rangle$$

allows us to replace the nonterminal $\langle \text{wff} \rangle$ by $\langle \text{compound} \rangle$ when the well-formed formula we are seeking to generate is a compound formula. Successive applications of these three productions to the nonterminal $\langle \text{wff} \rangle$ yield strings such as

$$\langle \text{atom} \rangle, \quad \langle \text{compound} \rangle, \quad (\langle \text{atom} \rangle), \quad ((((\langle \text{compound} \rangle)))).$$

The sequence of steps that transform $\langle \text{wff} \rangle$ into $((((\langle \text{compound} \rangle))))$ may be presented as follows:

$$\begin{aligned} \langle \text{wff} \rangle &\Rightarrow (\langle \text{wff} \rangle) \\ &\Rightarrow ((\langle \text{wff} \rangle)) \\ &\Rightarrow ((((\langle \text{wff} \rangle)))) \\ &\Rightarrow ((((\langle \text{compound} \rangle)))) \end{aligned}$$

The first three steps apply the production $\langle \text{wff} \rangle \to (\langle \text{wff} \rangle)$ to the nonterminal $\langle \text{wff} \rangle$ in the relevant formula, replacing this nonterminal by $(\langle \text{wff} \rangle)$. The final step applies the production $\langle \text{wff} \rangle \to \langle \text{compound} \rangle$ to the nonterminal $\langle \text{wff} \rangle$ in the penultimate formula, replacing this nonterminal by $\langle \text{compound} \rangle$. (Where the symbol $\Rightarrow$ occurs between two strings, this indicates that the first string can be transformed into the second string on applying one of the productions of the grammar to a single nonterminal occuring in the first string to replace that nonterminal by the appropriate string.) We write

$$\langle \text{wff} \rangle \stackrel{*}{\Rightarrow} ((((\langle \text{compound} \rangle))))$$

to indicate that the string on the left can be transformed into the string on the right through the successive application of a finite number of productions belonging to the grammar. (In general, where the symbol $\stackrel{*}{\Rightarrow}$ is placed between two strings, this indicates either that the first string is identical to the second, or else that the first string can be transformed into the second through the successive application of a finite number of productions belonging to the relevant grammar.)

The three productions which we can apply to the nonterminal ⟨wff⟩ can be specified more compactly in *Backus-Naur form* by means of the following:

$$\langle \text{wff} \rangle \rightarrow (\langle \text{wff} \rangle) \mid \langle \text{atom} \rangle \mid \langle \text{compound} \rangle$$

This indicates that the nonterminal ⟨wff⟩ occurring on the left hand side may be replaced by any one of a list of alternatives presented on the right hand side. The meta-character | is used to separate the alternatives within this list.

We next consider the productions for producing atomic formulae. Any atomic formula represents either a Boolean constant or a Boolean variable. Moreover the Boolean constants are **T** and **F**. This leads us to introduce the following four productions:

$$\begin{aligned} \langle \text{atom} \rangle &\rightarrow \langle \text{constant} \rangle \\ \langle \text{atom} \rangle &\rightarrow \langle \text{variable} \rangle \\ \langle \text{constant} \rangle &\rightarrow \mathbf{T} \\ \langle \text{constant} \rangle &\rightarrow \mathbf{F} \end{aligned}$$

These four productions may be specified in Backus-Naur form by the following:

$$\begin{aligned} \langle \text{atom} \rangle &\rightarrow \langle \text{constant} \rangle \mid \langle \text{variable} \rangle \\ \langle \text{constant} \rangle &\rightarrow \mathbf{T} \mid \mathbf{F} \end{aligned}$$

Using these productions we find that

$$\langle \text{atom} \rangle \Rightarrow \langle \text{constant} \rangle \Rightarrow \mathbf{T}, \quad \langle \text{atom} \rangle \Rightarrow \langle \text{constant} \rangle \Rightarrow \mathbf{F},$$

and thus

$$\langle \text{atom} \rangle \stackrel{*}{\Rightarrow} \mathbf{T}, \quad \langle \text{atom} \rangle \stackrel{*}{\Rightarrow} \mathbf{F}.$$

We also need to specify the productions that transform the nonterminal ⟨variable⟩ into the formulae

$$p, q, r, s, p', q', r', s', p'', q'', r'', s'', p''', q''', r''', s''', \ldots .$$

These transformations may be accomplished using the following productions:

$$\begin{aligned} \langle \text{variable} \rangle &\rightarrow \langle \text{variable} \rangle' \\ \langle \text{variable} \rangle &\rightarrow \langle \text{letter} \rangle \\ \langle \text{letter} \rangle &\rightarrow p \\ \langle \text{letter} \rangle &\rightarrow q \\ \langle \text{letter} \rangle &\rightarrow r \\ \langle \text{letter} \rangle &\rightarrow s \end{aligned}$$

These productions can be specified in Backus-Naur form as follows:

$$\begin{aligned} \langle \text{variable} \rangle &\rightarrow \langle \text{letter} \rangle \mid \langle \text{variable} \rangle' \\ \langle \text{letter} \rangle &\rightarrow p \mid q \mid r \mid s \end{aligned}$$

For example, the formula p''' is generated from the nonterminal $\langle \text{variable} \rangle$ by the following sequence of transformations:

$$\langle \text{variable} \rangle \Rightarrow \langle \text{variable} \rangle' \Rightarrow \langle \text{variable} \rangle'' \Rightarrow \langle \text{variable} \rangle''' \Rightarrow \langle \text{letter} \rangle''' \Rightarrow p'''.$$

The first three transformations use the production $\langle \text{variable} \rangle \rightarrow \langle \text{variable} \rangle'$, the fourth uses the production $\langle \text{variable} \rangle \rightarrow \langle \text{letter} \rangle$, and the final transformation uses the production $\langle \text{letter} \rangle \rightarrow p$. Thus $\langle \text{variable} \rangle \stackrel{*}{\Rightarrow} p'''$. The formulae representing Boolean constants and variables can now all be generated from the nonterminal $\langle \text{wff} \rangle$. For example, $\langle \text{wff} \rangle \stackrel{*}{\Rightarrow} q'$, since

$$\langle \text{wff} \rangle \Rightarrow \langle \text{atom} \rangle \Rightarrow \langle \text{variable} \rangle \Rightarrow \langle \text{variable} \rangle' \Rightarrow \langle \text{letter} \rangle' \Rightarrow q'.$$

Our grammar now has productions to generate any atomic formula. We now need to add productions that will generate compound formulae. A compound formula is either the negation of some other well-formed formula, or the conjunction of two well-formed formulae, or the disjunction of two such formulae. This leads us to introduce the productions

$$\begin{aligned} \langle \text{compound} \rangle &\rightarrow \langle \text{negation} \rangle \\ \langle \text{compound} \rangle &\rightarrow \langle \text{conjunction} \rangle \\ \langle \text{compound} \rangle &\rightarrow \langle \text{disjunction} \rangle \end{aligned}$$

which may be specified in Backus-Naur form as

$$\langle \text{compound} \rangle \rightarrow \langle \text{negation} \rangle \mid \langle \text{conjunction} \rangle \mid \langle \text{disjunction} \rangle$$

Let us consider negations. If F is any well-formed formula, then $\neg(F)$ is a well-formed formula that represents the negation of the formula F. Also we

shall regard $\neg F$ as being a well-formed formula representing the negation of F in the cases when F is atomic or when F is itself a negation. However we will adopt the convention that the negation operation $\neg$ has higher precedence than either $\wedge$ or $\vee$. Thus if F is of the form $G \wedge H$ then $\neg G \wedge H$ will be equivalent to $(\neg G) \wedge H$ and will not represent the negation $\neg(G \wedge H)$ of F. Similarly $\neg G \vee H$ is to be interpreted as $(\neg G) \vee H$, and is not equivalent to $\neg(G \vee H)$. Therefore we only denote the negation of a well-formed formula F by $\neg F$ in the cases when F is atomic or a negation, and not in the cases when F is a conjunction or disjunction. We therefore introduce the following productions into our grammar:

$$\begin{aligned} \langle \text{negation} \rangle &\rightarrow \neg \langle \text{nf} \rangle \\ \langle \text{nf} \rangle &\rightarrow \langle \text{atom} \rangle \\ \langle \text{nf} \rangle &\rightarrow \langle \text{negation} \rangle \\ \langle \text{nf} \rangle &\rightarrow (\langle \text{wff} \rangle) \end{aligned}$$

These productions are specified in Backus-Naur form as follows:

$$\begin{aligned} \langle \text{negation} \rangle &\rightarrow \neg \langle \text{nf} \rangle \\ \langle \text{nf} \rangle &\rightarrow \langle \text{atom} \rangle \mid \langle \text{negation} \rangle \mid (\langle \text{wff} \rangle) \end{aligned}$$

Next let us consider conjunctions. If G and H are well-formed formulae then $(G) \wedge (H)$ is a well-formed formula representing the conjunction of G and H. We may replace $(G) \wedge (H)$ by $G \wedge (H)$ without introducing any ambiguity in the cases when G is atomic, the negation of a well-formed formula or a conjunction of well-formed formulae. (Our rules of precedence ensure that $\neg G \wedge H$ is interpreted as $(\neg G) \wedge H$ and not as $\neg(G \wedge H)$, since we regard negation has having higher precedence than conjunction.) We shall not allow ourselves to replace $(G) \wedge (H)$ by $G \wedge (H)$ when G is a disjunction of well-formed formulae. Similarly we may replace $(G) \wedge (H)$ and $G \wedge (H)$ by $(G) \wedge H$ and $G \wedge H$ respectively when the formula H is an atomic formula or the negation of a well-formed formula, or a conjunction of well-formed formulae, but not when H is a disjunction of well-formed formulae. These considerations are respected if we introduce the productions

$$\begin{aligned} \langle \text{conjunction} \rangle &\rightarrow \langle \text{cf} \rangle \wedge \langle \text{cf} \rangle \\ \langle \text{cf} \rangle &\rightarrow \langle \text{atom} \rangle \\ \langle \text{cf} \rangle &\rightarrow \langle \text{negation} \rangle \\ \langle \text{cf} \rangle &\rightarrow \langle \text{conjunction} \rangle \\ \langle \text{cf} \rangle &\rightarrow (\langle \text{wff} \rangle) \end{aligned}$$

which may be specified in Backus-Naur form as follows:

$$\begin{aligned} \langle \text{conjunction} \rangle &\rightarrow \langle \text{cf} \rangle \wedge \langle \text{cf} \rangle \\ \langle \text{cf} \rangle &\rightarrow \langle \text{atom} \rangle \mid \langle \text{negation} \rangle \mid \langle \text{conjunction} \rangle \mid (\langle \text{wff} \rangle) \end{aligned}$$

With these productions we find, for example, that

$$\begin{aligned} \langle \text{conjunction} \rangle &\Rightarrow \langle \text{cf} \rangle \wedge \langle \text{cf} \rangle \\ &\Rightarrow \langle \text{atom} \rangle \wedge \langle \text{cf} \rangle \\ &\Rightarrow \langle \text{atom} \rangle \wedge \langle \text{negation} \rangle \\ &\Rightarrow \langle \text{atom} \rangle \wedge \neg \langle \text{nf} \rangle \\ &\Rightarrow \langle \text{atom} \rangle \wedge \neg \langle \text{atom} \rangle \end{aligned}$$

and

$$\begin{aligned} \langle \text{conjunction} \rangle &\Rightarrow \langle \text{cf} \rangle \wedge \langle \text{cf} \rangle \\ &\Rightarrow \langle \text{atom} \rangle \wedge \langle \text{cf} \rangle \\ &\Rightarrow \langle \text{atom} \rangle \wedge \langle \text{conjunction} \rangle \\ &\Rightarrow \langle \text{atom} \rangle \wedge \langle \text{cf} \rangle \wedge \langle \text{cf} \rangle \\ &\Rightarrow \langle \text{atom} \rangle \wedge \langle \text{atom} \rangle \wedge \langle \text{cf} \rangle \\ &\Rightarrow \langle \text{atom} \rangle \wedge \langle \text{atom} \rangle \wedge \langle \text{atom} \rangle. \end{aligned}$$

We can then apply further productions in order to obtain formulae such as $p \wedge \neg q$ and $p' \wedge p'' \wedge p'''$.

Finally we have to specify the productions for handling disjunction. These are analogous to those for conjunctions, and are the following:—

$$\begin{aligned} \langle \text{disjunction} \rangle &\rightarrow \langle \text{df} \rangle \vee \langle \text{df} \rangle \\ \langle \text{df} \rangle &\rightarrow \langle \text{atom} \rangle \\ \langle \text{df} \rangle &\rightarrow \langle \text{negation} \rangle \\ \langle \text{df} \rangle &\rightarrow \langle \text{disjunction} \rangle \\ \langle \text{df} \rangle &\rightarrow (\langle \text{wff} \rangle) \end{aligned}$$

These can be presented in Backus-Naur form as follows:

$$\begin{aligned} \langle \text{disjunction} \rangle &\rightarrow \langle \text{df} \rangle \vee \langle \text{df} \rangle \\ \langle \text{df} \rangle &\rightarrow \langle \text{atom} \rangle \mid \langle \text{negation} \rangle \mid \langle \text{disjunction} \rangle \mid (\langle \text{wff} \rangle) \end{aligned}$$

With these productions we can generate formulae such as $p \vee (q \wedge r)$ by applying productions successively as follows:

$$\begin{aligned}
\langle\text{disjunction}\rangle &\Rightarrow \langle\text{df}\rangle \vee \langle\text{df}\rangle \\
&\Rightarrow \langle\text{atom}\rangle \vee \langle\text{df}\rangle \\
&\Rightarrow \langle\text{atom}\rangle \vee (\langle\text{wff}\rangle) \\
&\Rightarrow \langle\text{atom}\rangle \vee (\langle\text{compound}\rangle) \\
&\Rightarrow \langle\text{atom}\rangle \vee (\langle\text{conjunction}\rangle) \\
&\Rightarrow \langle\text{atom}\rangle \vee (\langle\text{cf}\rangle \wedge \langle\text{cf}\rangle) \\
&\Rightarrow \langle\text{atom}\rangle \vee (\langle\text{atom}\rangle \wedge \langle\text{cf}\rangle) \\
&\Rightarrow \langle\text{atom}\rangle \vee (\langle\text{atom}\rangle \wedge \langle\text{atom}\rangle)
\end{aligned}$$

This completes our construction of a context-free grammar that generates the well-formed formulae of the Propositional Calculus.

This grammar is specified in Backus-Naur form by the following:—

$$\begin{aligned}
\langle\text{wff}\rangle &\rightarrow (\langle\text{wff}\rangle) \mid \langle\text{atom}\rangle \mid \langle\text{compound}\rangle \\
\langle\text{atom}\rangle &\rightarrow \langle\text{constant}\rangle \mid \langle\text{variable}\rangle \\
\langle\text{compound}\rangle &\rightarrow \langle\text{negation}\rangle \mid \langle\text{conjunction}\rangle \mid \langle\text{disjunction}\rangle \\
\langle\text{negation}\rangle &\rightarrow \neg\langle\text{nf}\rangle \\
\langle\text{nf}\rangle &\rightarrow \langle\text{atom}\rangle \mid \langle\text{negation}\rangle \mid (\langle\text{wff}\rangle) \\
\langle\text{conjunction}\rangle &\rightarrow \langle\text{cf}\rangle \wedge \langle\text{cf}\rangle \\
\langle\text{cf}\rangle &\rightarrow \langle\text{atom}\rangle \mid \langle\text{negation}\rangle \mid \langle\text{conjunction}\rangle \mid (\langle\text{wff}\rangle) \\
\langle\text{disjunction}\rangle &\rightarrow \langle\text{df}\rangle \vee \langle\text{df}\rangle \\
\langle\text{df}\rangle &\rightarrow \langle\text{atom}\rangle \mid \langle\text{negation}\rangle \mid \langle\text{disjunction}\rangle \mid (\langle\text{wff}\rangle) \\
\langle\text{constant}\rangle &\rightarrow \mathbf{T} \mid \mathbf{F} \\
\langle\text{variable}\rangle &\rightarrow \langle\text{letter}\rangle \mid \langle\text{variable}\rangle' \\
\langle\text{letter}\rangle &\rightarrow p \mid q \mid r \mid s
\end{aligned}$$

The following are the productions of this grammar:—

$$\begin{aligned}
\langle\text{wff}\rangle &\rightarrow (\langle\text{wff}\rangle) \\
\langle\text{wff}\rangle &\rightarrow \langle\text{atom}\rangle \\
\langle\text{wff}\rangle &\rightarrow \langle\text{compound}\rangle \\
\langle\text{atom}\rangle &\rightarrow \langle\text{constant}\rangle \\
\langle\text{atom}\rangle &\rightarrow \langle\text{variable}\rangle \\
\langle\text{compound}\rangle &\rightarrow \langle\text{negation}\rangle
\end{aligned}$$

$$
\begin{aligned}
\langle \text{compound} \rangle &\to \langle \text{conjunction} \rangle \\
\langle \text{compound} \rangle &\to \langle \text{disjunction} \rangle \\
\langle \text{negation} \rangle &\to \neg \langle \text{nf} \rangle \\
\langle \text{nf} \rangle &\to \langle \text{atom} \rangle \\
\langle \text{nf} \rangle &\to \langle \text{negation} \rangle \\
\langle \text{nf} \rangle &\to (\langle \text{wff} \rangle) \\
\langle \text{conjunction} \rangle &\to \langle \text{cf} \rangle \wedge \langle \text{cf} \rangle \\
\langle \text{cf} \rangle &\to \langle \text{atom} \rangle \\
\langle \text{cf} \rangle &\to \langle \text{negation} \rangle \\
\langle \text{cf} \rangle &\to \langle \text{conjunction} \rangle \\
\langle \text{cf} \rangle &\to (\langle \text{wff} \rangle) \\
\langle \text{disjunction} \rangle &\to \langle \text{df} \rangle \vee \langle \text{df} \rangle \\
\langle \text{df} \rangle &\to \langle \text{atom} \rangle \\
\langle \text{df} \rangle &\to \langle \text{negation} \rangle \\
\langle \text{df} \rangle &\to \langle \text{disjunction} \rangle \\
\langle \text{df} \rangle &\to (\langle \text{wff} \rangle) \\
\langle \text{constant} \rangle &\to \mathbf{T} \\
\langle \text{constant} \rangle &\to \mathbf{F} \\
\langle \text{variable} \rangle &\to \langle \text{variable} \rangle' \\
\langle \text{variable} \rangle &\to \langle \text{letter} \rangle \\
\langle \text{letter} \rangle &\to p \\
\langle \text{letter} \rangle &\to q \\
\langle \text{letter} \rangle &\to r \\
\langle \text{letter} \rangle &\to s
\end{aligned}
$$

The well-formed formulae of the Propositional Calculus are those words F over the alphabet

$$\{\neg, \wedge, \vee, (,), \mathbf{T}, \mathbf{F}, p, q, r, s, '\}$$

for which $\langle \text{wff} \rangle \stackrel{*}{\Rightarrow} F$.

Example We verify that the formula

$$(p \wedge r') \vee ((\neg r'') \wedge q \wedge \neg r''')$$

is a well-formed formula of the Propositional Calculus. This formula may be obtained from the nonterminal $\langle \text{wff} \rangle$ by applying successive productions as

follows:

$$
\begin{aligned}
\langle \text{wff} \rangle &\Rightarrow \langle \text{compound} \rangle \\
&\Rightarrow \langle \text{disjunction} \rangle \\
&\Rightarrow \langle \text{df} \rangle \vee \langle \text{df} \rangle \\
&\Rightarrow (\langle \text{wff} \rangle) \vee \langle \text{df} \rangle \\
&\Rightarrow (\langle \text{compound} \rangle) \vee \langle \text{df} \rangle \\
&\Rightarrow (\langle \text{conjunction} \rangle) \vee \langle \text{df} \rangle \\
&\Rightarrow (\langle \text{cf} \rangle \wedge \langle \text{cf} \rangle) \vee \langle \text{df} \rangle \\
&\Rightarrow (\langle \text{atom} \rangle \wedge \langle \text{cf} \rangle) \vee \langle \text{df} \rangle \\
&\Rightarrow (\langle \text{variable} \rangle \wedge \langle \text{cf} \rangle) \vee \langle \text{df} \rangle \\
&\Rightarrow (\langle \text{letter} \rangle \wedge \langle \text{cf} \rangle) \vee \langle \text{df} \rangle \\
&\Rightarrow (p \wedge \langle \text{cf} \rangle) \vee \langle \text{df} \rangle \\
&\Rightarrow (p \wedge \langle \text{atom} \rangle) \vee \langle \text{df} \rangle \\
&\Rightarrow (p \wedge \langle \text{variable} \rangle) \vee \langle \text{df} \rangle \\
&\Rightarrow (p \wedge \langle \text{variable} \rangle') \vee \langle \text{df} \rangle \\
&\Rightarrow (p \wedge \langle \text{letter} \rangle') \vee \langle \text{df} \rangle \\
&\Rightarrow (p \wedge r') \vee \langle \text{df} \rangle \\
&\Rightarrow (p \wedge r') \vee (\langle \text{wff} \rangle) \\
&\Rightarrow (p \wedge r') \vee (\langle \text{compound} \rangle) \\
&\Rightarrow (p \wedge r') \vee (\langle \text{conjunction} \rangle) \\
&\Rightarrow (p \wedge r') \vee (\langle \text{cf} \rangle \wedge \langle \text{cf} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\langle \text{wff} \rangle) \wedge \langle \text{cf} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\langle \text{compound} \rangle) \wedge \langle \text{cf} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\langle \text{negation} \rangle) \wedge \langle \text{cf} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\neg \langle \text{nf} \rangle) \wedge \langle \text{cf} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\neg \langle \text{atom} \rangle) \wedge \langle \text{cf} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\neg \langle \text{variable} \rangle) \wedge \langle \text{cf} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\neg \langle \text{variable} \rangle') \wedge \langle \text{cf} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\neg \langle \text{variable} \rangle'') \wedge \langle \text{cf} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\neg \langle \text{letter} \rangle'') \wedge \langle \text{cf} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\neg r'') \wedge \langle \text{cf} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\neg r'') \wedge \langle \text{conjunction} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\neg r'') \wedge \langle \text{cf} \rangle \wedge \langle \text{cf} \rangle) \\
&\Rightarrow (p \wedge r') \vee ((\neg r'') \wedge \langle \text{atom} \rangle \wedge \langle \text{cf} \rangle)
\end{aligned}
$$

$$
\begin{aligned}
&\Rightarrow\ (p \wedge r') \vee ((\neg r'') \wedge \langle \text{variable} \rangle \wedge \langle \text{cf} \rangle)\\
&\Rightarrow\ (p \wedge r') \vee ((\neg r'') \wedge \langle \text{letter} \rangle \wedge \langle \text{cf} \rangle)\\
&\Rightarrow\ (p \wedge r') \vee ((\neg r'') \wedge q \wedge \langle \text{cf} \rangle)\\
&\Rightarrow\ (p \wedge r') \vee ((\neg r'') \wedge q \wedge \langle \text{negation} \rangle)\\
&\Rightarrow\ (p \wedge r') \vee ((\neg r'') \wedge q \wedge \neg\langle \text{nf} \rangle)\\
&\Rightarrow\ (p \wedge r') \vee ((\neg r'') \wedge q \wedge \neg\langle \text{atom} \rangle)\\
&\Rightarrow\ (p \wedge r') \vee ((\neg r'') \wedge q \wedge \neg\langle \text{variable} \rangle)\\
&\Rightarrow\ (p \wedge r') \vee ((\neg r'') \wedge q \wedge \neg\langle \text{variable} \rangle')\\
&\Rightarrow\ (p \wedge r') \vee ((\neg r'') \wedge q \wedge \neg\langle \text{variable} \rangle'')\\
&\Rightarrow\ (p \wedge r') \vee ((\neg r'') \wedge q \wedge \neg\langle \text{variable} \rangle''')\\
&\Rightarrow\ (p \wedge r') \vee ((\neg r'') \wedge q \wedge \neg\langle \text{letter} \rangle''')\\
&\Rightarrow\ (p \wedge r') \vee ((\neg r'') \wedge q \wedge \neg r''')
\end{aligned}
$$

Remark The grammar we have constructed to describe the well-formed formulae of the Propositional Calculus generates formulae such as $(((p \wedge q)))$ and $p \vee ((((((q))))))$ which are unambiguous but which contain superfluous parentheses which serve no useful purpose. It is possible to modify the grammar to ensure that the modified grammar does not generate formulae with such superfluous parentheses. In particular we can ensure that a formula generated by the modified grammar is never enclosed within parentheses, that no Boolean constant or variable within the formula is enclosed by itself within parentheses, and that no subformula is enclosed by itself within two or more sets of parentheses. This modified grammar is expressed in Backus-Naur form as follows:—

$$
\begin{aligned}
\langle \text{wff} \rangle &\ \rightarrow\ \langle \text{atom} \rangle \mid \langle \text{compound} \rangle\\
\langle \text{atom} \rangle &\ \rightarrow\ \langle \text{constant} \rangle \mid \langle \text{variable} \rangle\\
\langle \text{compound} \rangle &\ \rightarrow\ \langle \text{negation} \rangle \mid \langle \text{conjunction} \rangle \mid \langle \text{disjunction} \rangle\\
\langle \text{negation} \rangle &\ \rightarrow\ \neg\langle \text{nf} \rangle\\
\langle \text{nf} \rangle &\ \rightarrow\ \langle \text{atom} \rangle \mid \langle \text{negation} \rangle \mid (\langle \text{compound} \rangle)\\
\langle \text{conjunction} \rangle &\ \rightarrow\ \langle \text{cf} \rangle \wedge \langle \text{cf} \rangle\\
\langle \text{cf} \rangle &\ \rightarrow\ \langle \text{atom} \rangle \mid \langle \text{negation} \rangle \mid \langle \text{conjunction} \rangle \mid (\langle \text{compound} \rangle)\\
\langle \text{disjunction} \rangle &\ \rightarrow\ \langle \text{df} \rangle \vee \langle \text{df} \rangle\\
\langle \text{df} \rangle &\ \rightarrow\ \langle \text{atom} \rangle \mid \langle \text{negation} \rangle \mid \langle \text{disjunction} \rangle \mid (\langle \text{compound} \rangle)\\
\langle \text{constant} \rangle &\ \rightarrow\ \mathbf{T} \mid \mathbf{F}\\
\langle \text{variable} \rangle &\ \rightarrow\ \langle \text{letter} \rangle \mid \langle \text{variable} \rangle'\\
\langle \text{letter} \rangle &\ \rightarrow\ p \mid q \mid r \mid s
\end{aligned}
$$

5.4 Context-Free Grammars

We have discussed examples of context-free grammars. We now present and discuss a formal definition of such grammars.

Definition A *context-free grammar* $(V, A, \langle S \rangle, P)$ consists of a finite set V, a subset A of V, an element $\langle S \rangle$ of $V \setminus A$, and a finite subset P of $(V \setminus A) \times V^*$.

Let $(V, A, \langle S \rangle, P)$ be a context-free grammar. The elements of A are referred to as *terminals*. Let $N = V \setminus A$. The elements of N are referred to as *nonterminals*. The nonterminal $\langle S \rangle$ is the *start symbol*. The set N of nonterminals is non-empty since $\langle S \rangle \in N$.

The finite set P specifies the *productions* of the grammar. An element of P is an ordered pair of the form $(\langle T \rangle, w)$ where $\langle T \rangle \in N$ is a nonterminal and $w \in V^*$ is a word over the alphabet V (i.e, a finite string, where each constituent of the string is either a terminal or a nonterminal). We denote by

$$\langle T \rangle \to w$$

the production specified by an ordered pair $(\langle T \rangle, w)$ belonging to the set P.

Definition Let w' and w'' be words over the alphabet V. We say that w' *directly yields* w'' if there exist words u and v over the alphabet V and a production $\langle T \rangle \to w$ of the grammar such that $w' = u\langle T \rangle v$ and $w'' = uwv$. (Either or both of the words u and v may be the empty word.)

We see therefore that a word w' over the alphabet V directly yields another such word w'' if and only if there exists a production $\langle T \rangle \to w$ in the grammar such that w'' may be obtained from w' by replacing a single occurrence of the nonterminal $\langle T \rangle$ within w' by the word w. If the word w' directly yields w'', then we denote this fact by writing

$$w' \Rightarrow w''.$$

Definition Let w' and w'' be words over the alphabet V. We say that w' *yields* w'' if either $w' = w''$ or else there exist words $w_0, w_1, \ldots, w_n$ over the alphabet V such that $w_0 = w'$, $w_n = w''$ and $w_{i-1} \Rightarrow w_i$ for all integers i between 1 and n.

If the word w' yields w'', then we denote this fact by writing

$$w' \stackrel{*}{\Rightarrow} w''.$$

Definition Let $(V, A, \langle S\rangle, P)$ be a context-free grammar. The *language* generated by this grammar is the subset L of A^* defined by

$$L = \{w \in A^* : \langle S\rangle \stackrel{*}{\Rightarrow} w\}.$$

We see therefore that the language L generated by a context-free grammar $(V, A, \langle S\rangle, P)$ consists of the set of all finite strings consisting entirely of terminals that may be obtained from the start symbol $\langle S\rangle$ by applying a finite sequence of productions of the grammar, where the application of a production causes a single nonterminal to be replaced by the string in V^* specified by that production.

5.5 Phrase Structure Grammars

There is a class of formal grammars that includes all context-free grammars. The grammars of this more general type are known as *phase structure grammars*

Definition A *phrase structure grammar* $(V, A, \langle S\rangle, P)$ consists of a finite set V, a subset A of V, an element $\langle S\rangle$ of $V \setminus A$, and a finite subset P of $(V^* \setminus A^*) \times V^*$.

As in the case of context-free grammars, the elements of A are referred to as *terminals*, the elements of $V \setminus A$ are referred to as *nonterminals*, the nonterminal $\langle S\rangle$ is the *start symbol* and the elements of P specify the *productions* of the grammar. The production specified by an element (r, w) of P is denoted by $r \to w$. However the left hand side r of a production $r \to w$ in a phrase structure grammar need not consist solely of a single nonterminal, but may be a finite string r of elements of V^*, provided that this string r contains at least one nonterminal. (Note that V^* denotes the set of all finite words over the alphabet V whose elements are terminals and nonterminals, A^* denotes the set of all finite words consisting entirely of terminals, and thus $V^* \setminus A^*$ denotes the set of all finite words belonging to V^* which contain at least one nonterminal.)

Definition Let w' and w'' be words over the alphabet V. We say that w' *directly yields* w'' if there exist words u and v over the alphabet V and a production $r \to w$ such that $w' = urv$ and $w'' = uwv$. (Either or both of the words u and v may be the empty word.)

We see therefore that a word w' over the alphabet V directly yields another such word w'' if and only if there exists a production $r \to w$ in the

grammar such that w'' may be obtained from w' by replacing a single occurrence of r as a substring of w' by the string w. If the word w' directly yields w'', then we denote this fact by writing

$$w_1 \Rightarrow w_2.$$

Definition Let w' and w'' be words over the alphabet V. We say that w' *yields* w'' if either $w' = w''$ or else there exist words $w_0, w_1, \ldots, w_n$ over the alphabet w such that $w_0 = w'$, $w_n = w''$ and $w_{i-1} \Rightarrow w_i$ for all integers i between 1 and n.

If the word w' yields w'', then we denote this fact by writing

$$w' \stackrel{*}{\Rightarrow} w''.$$

Definition Let $(V, A, \langle S\rangle, P)$ be a phrase structure grammar. The *language* generated by this grammar is the subset L of A^* defined by

$$L = \{w \in A^* : \langle S\rangle \stackrel{*}{\Rightarrow} w\}.$$

5.6 Regular Languages

Definition Let A be a finite set, and let A^* be the set of words over the alphabet A. A subset L of A^* is said to be a *regular language* over the alphabet A if $L = L_m$ for some finite sequence $L_1, L_2, \ldots, L_m$ of subsets of A^* with the property that, for each integer i between 1 and m, the set L_i satisfies at least one of the following conditions:—

(i) L_i is a finite set;

(ii) $L_i = L_j^*$ for some integer j satisfying $1 \leq j < i$;

(iii) $L_i = L_j \circ L_k$ for some integers j and k satisfying $1 \leq j < i$, $1 \leq k < i$;

(iv) $L_i = L_j \cup L_k$ for some integers j and k satisfying $1 \leq j < i$, $1 \leq k < i$.

(Here $L_j \circ L_k$ denotes the set of all words over the alphabet A that are concatenations of the form $w'w''$ with $w' \in L_j$ and $w'' \in L_k$.)

Let A be a finite set, and let A^* be the set of all words over the alphabet A. The regular languages over the alphabet A constitute the smallest collection $\mathcal{C}$ of subsets of A^* which satisfies the following properties:—

(i) all finite subsets of A^* belong to $\mathcal{C}$;

(ii) if M is a subset of A^* belonging to $\mathcal{C}$ then so is M^*;

(iii) if M and N are subsets of A^* belonging to $\mathcal{C}$ then so is $M \circ N$.

(iv) if M and N are subsets of A^* belonging to $\mathcal{C}$ then so is $M \cup N$.

Indeed the collection of regular languages over the alphabet A has all four properties. Moreover any collection of languages over the alphabet A with these properties includes all regular languages. Indeed if L is a regular language then $L = L_m$ for some finite sequence $L_1, L_2, \ldots, L_m$ of subsets of A^* which each set L_i is either a finite set, or of the form L_j^* for some set L_j with $j < i$, or of the form $L_j \circ L_k$ for some sets L_j and L_k with $j < i$ and $k < i$, or of the form $L_j \cup L_k$ for some sets L_j and L_k with $j < i$ and $k < i$. It follows from this that if a collection $\mathcal{C}$ of subsets of A^* with the four properties given above contains L_j for all integers j satisfying $1 \leq j < i$ then it also contains L_i. Therefore all of the sets $L_1, L_2, \ldots, L_m$ must belong to the collection $\mathcal{C}$, and, in particular, the regular language L must belong to $\mathcal{C}$.

Example Let L be the set of decimal representations of integers. We give each integer a unique decimal representation in L, so that the decimal representation of any positive integer in L begins with a non-zero digit, the decimal representation of zero is '0', and the decimal representation of any negative number begins with a minus sign followed by a non-zero digit. The set L is a language over the alphabet A, where

$$A = \{-, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}.$$

Moreover $L = L_9$, where

$$\begin{aligned}
L_1 &= \{0\}, \\
L_2 &= \{1, 2, 3, 4, 5, 6, 7, 8, 9\}, \\
L_3 &= L_1 \cup L_2, \\
L_4 &= L_3^*, \\
L_5 &= L_2 \circ L_4, \\
L_6 &= \{-\}, \\
L_7 &= L_6 \circ L_5, \\
L_8 &= L_5 \cup L_7, \\
L_9 &= L_8 \cup L_1.
\end{aligned}$$

Note that L_3 is the set of decimal digits, L_4 is the set of finite strings of decimal digits (including the empty string), L_5 is the set of decimal representations of positive integers, L_7 is the set of decimal representations of

negative integers, and L_8 is the set of decimal representations of non-zero integers. We conclude that L is a regular language over the alphabet A.

The regular languages may be characterised as those languages that are generated by *regular grammars*. They may also be characterised as those languages that are recognized by *finite state acceptors*. We shall give below formal definitions of *regular grammars* and *finite state acceptors*.

5.7 Regular Grammars

Definition A context-free grammar is said to be a *regular grammar* if every production is of one of the three forms

$$\begin{array}{lrcl} \text{(i)} & \langle A \rangle & \rightarrow & b\langle B \rangle, \\ \text{(ii)} & \langle A \rangle & \rightarrow & b, \\ \text{(iii)} & \langle A \rangle & \rightarrow & \varepsilon, \end{array}$$

where $\langle A \rangle$ and $\langle B \rangle$ represent nonterminals, b represents a terminal, and ε denotes the empty word. A regular grammar is said to be in *normal form* if all its productions are of types (i) and (iii).

Lemma 5.2 *Any language generated by a regular grammar may be generated by a regular grammar in normal form.*

Proof Let L be a language over an alphabet A, and let $(V, A, \langle S \rangle, P)$ be a regular grammar generating the language L. We may construct a new regular grammar in normal form by first adding a nonterminal $\langle F \rangle$ that does not already belong to V, and then replacing any production which is of the form $\langle A \rangle \rightarrow b$ for some nonterminal $\langle A \rangle$ and terminal b by the pair of productions $\langle A \rangle \rightarrow b\langle F \rangle$ and $\langle F \rangle \rightarrow \varepsilon$. (Indeed the replacement of $\langle A \rangle$ by b in any word may be accomplished in two steps, by first replacing $\langle A \rangle$ by $b\langle F \rangle$ and then replacing $\langle F \rangle$ by the empty word ε.) The resultant regular grammar is in normal form and generates the same language as the given regular grammar. ∎

Example Let L be the set of decimal representations of integers, in which the most significant digit of a non-zero integer is non-zero, and in which zero is represented by '0'. L is a regular language over the alphabet A, where

$$A = \{-, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}.$$

This language is generated by the regular grammar whose nonterminals are $\langle S\rangle$, $\langle A\rangle$, $\langle B\rangle$ and whose productions are

$$\begin{array}{lcl}
\langle S\rangle & \to & -\langle A\rangle \\
\langle S\rangle & \to & 0 \\
\langle S\rangle & \to & 1\langle B\rangle \\
\langle S\rangle & \to & 2\langle B\rangle \\
 & \vdots & \\
\langle S\rangle & \to & 9\langle B\rangle \\
\langle A\rangle & \to & 1\langle B\rangle \\
\langle A\rangle & \to & 2\langle B\rangle \\
 & \vdots & \\
\langle A\rangle & \to & 9\langle B\rangle \\
\langle B\rangle & \to & 0\langle B\rangle \\
\langle B\rangle & \to & 1\langle B\rangle \\
\langle B\rangle & \to & 2\langle B\rangle \\
 & \vdots & \\
\langle B\rangle & \to & 9\langle B\rangle \\
\langle B\rangle & \to & \varepsilon
\end{array}$$

This regular grammar is not in normal form, but to obtain a regular grammar in normal form it suffices to introduce a new nonterminal $\langle C\rangle$, and replace the production $\langle S\rangle \to 0$ by the two productions

$$\begin{array}{lcl}
\langle S\rangle & \to & 0\langle C\rangle \\
\langle C\rangle & \to & \varepsilon
\end{array}$$

5.8 Finite State Acceptors

Definition A *finite state acceptor* (S, A, i, t, F) consists of finite sets S and A, an element i of S, a function $t\colon S \times A \to S$ from $S \times A$ to S and a subset F of S. The set S is the *set of states*, the set A is the *input alphabet*, the element i of S is the *starting state*, the function $t\colon S \times A \to S$ is the *transition mapping* and F is the set of *finishing states*.

A finite state acceptor is a particular type of *finite state machine*. Such a machine has a finite number of internal states. Data is input discretely,

and each datum causes the machine either to remain in the same internal state or else to make a transition to some other internal state determined solely by the current state and the input datum. In a finite state acceptor (S, A, i, t, F) the set S represents of the internal states of the machine and is finite, and each datum is an element of the input alphabet A. The machine is initially in the starting state i. The transition function t specifies how the internal state of the machine changes on inputting a datum: if the machine is currently in state s, and if the input datum is a, then the internal state of the machine becomes s' where $s' = t(s, a)$. Any finite state acceptor determines a language L over the alphabet A, consisting of those words $a_1 a_2 \dots a_n$ which, when the elements $a_1, a_2, \dots, a_n$ of A are successively input, cause the machine initially in the starting state to end up in one of the finishing states specified by the subset F of S.

Definition Let (S, A, i, t, F) be a finite state acceptor, and let A^* denote the set of words over the input alphabet A. A word $a_1 a_2 \dots a_n$ of length n over the alphabet A is said to be *recognized* or *accepted* by the finite state acceptor if there are states $s_0, s_1, s_2, \dots, s_n$ belonging to S such that $s_0 = i$, $s_n \in F$, and $s_i = t(s_{i-1}, a_i)$ for each integer i between 1 and n.

Definition Let (S, A, i, t, F) be a finite state acceptor. A language L over the alphabet A is said to be *recognized* or *accepted* by the finite state acceptor if L is the set consisting of all words recognized by the finite state acceptor.

It can be proved that a language over some alphabet A is a regular language if and only if that language is recognized by some finite state acceptor with input alphabet A.

Example Let L be the set of decimal representations of integers, in which the most significant digit of a non-zero integer is non-zero, and in which zero is represented by '0'. L is a regular language over the alphabet A, where

$$A = \{-, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}.$$

Let $S = \{i, a, b, c, e\}$, let $F = \{b, c\}$, and let $t\colon S \times A \to S$ be the transition mapping determined by the following *transition table*:—

	-	0	1	2	3	4	5	6	7	8	9
i	a	c	b	b	b	b	b	b	b	b	b
a	e	e	b	b	b	b	b	b	b	b	b
b	e	b	b	b	b	b	b	b	b	b	b
c	e	e	e	e	e	e	e	e	e	e	e
e	e	e	e	e	e	e	e	e	e	e	e

(Here the value of $t(s, \alpha)$ for a state s and character α is listed in the row of the table labelled by s and the column labelled by α.) Then (S, A, i, t, F) is a finite state acceptor for the regular language L.