2 Foundations of Mathematics I

This chapter and the next comprise a potpourri of introductory topics drawn from the foundations of mathematics. These topics, largely chosen from mathematical logic and from the study of formal systems, are particularly important in putting the study of the computer on a more scientific basis. In fact, the salient mathematical ideas set the stage historically for the development and utilization of the electronic digital computer.

The notion of a function, or a mapping of objects of some "domain" into objects of some "range," underlies programming. Programs define algorithms for computing functions, and the methods for defining functions have implications to the design of programming languages; well-designed languages include features to facilitate the computation of functions. A hierarchy of functions exists, with functions of functions being at a higher level, and these notions have led to some specialized programming features in some advanced languages.

Formal systems are systems representing mathematical structures and in which symbols are manipulated in game-like fashion. These symbols denote elements of the structure being represented, and the results of these manipulations are translatable into properties of that structure. Once defined, a formal system concerns form rather than meaning. These ideas relate to the use of a computer in simulation. Also, the technique of *arithmetization* in logic, of casting a mathematical structure in terms of numbers and arithmetic operations, is in the same vein as the application of the computer to a multitude of situations where useful results are obtained by working within the domain of numbers.

The same "laws of thought" that were studied by Boole in his development of Boolean algebra are at the root of computer processes. This area of logic is essential both to the design of computer hardware and to the design of computer languages.

2.1 INTRODUCTION

The foundations of mathematics concern the recognition and study of the fundamental objects of mathematics (such as numbers, sets, points, lines, etc.), their basic properties and the use and limitations of the logical and mathematical methods we employ in building theories about them. The subject matter appropriate to these studies is vast, and we shall only touch very lightly on some aspects of it that are particularly relevant to the computer sciences. As we have indicated earlier, digital computers are mathematical machines. In any use of a computer at all we are computing some mathematical function—that which maps the input, interpreted as a number, or perhaps, as a string of numbers, into the output, similarly interpreted. It is, therefore, not surprising that the foundations of mathematics play a role in the study in depth of these devices.

2.2 THE MEANING AND EXTENSIONS OF FUNCTION

We have briefly discussed in Chapter 1 the notion of a function and shall now give a somewhat amplified treatment of this concept because it plays such a vital role in computer-related, and indeed in all, mathematics.

Let us consider, first, a situation where we are interested in the relationship between two things that have magnitude and can, therefore, assume numeric values. It is often convenient, although not necessary, to think of one of these things as being allowed to vary independently of the other and of the second as being determined, perhaps in some unknown way, by this independent variable. For example, consider the question of describing the volume of a sphere in terms of the radius. It is natural to think of the radius as the independent variable, or the *argument*, and of the volume as being the dependent variable. The correspondence between the radius and the volume is expressed by the equation

$$V = \tfrac{4}{3}\pi r^3.$$

The format of the equation clearly signals the independent variable r, on the right, and the dependent variable V, on the left of the equal sign, and it defines V as a function of r. A function then is a *correspondence* between two variables that associates with each given admissible value of one of the variables a unique value of the second variable. Using our earlier terminology (Chapter I), it is a *map* relating these variables (just as an ordinary map relates some part of the earth's surface to a configuration on a sheet of paper).

If we had used other names for the variables involved, say W for the volume and s for the radius, and written

$$W = \tfrac{4}{3}\pi s^3,$$

the function or correspondence between these two variables would be the same. As we learn in high school algebra this correspondence is often shown graphically, and the picture in Fig. 2.1 might be used to depict the correspondence between r and V.

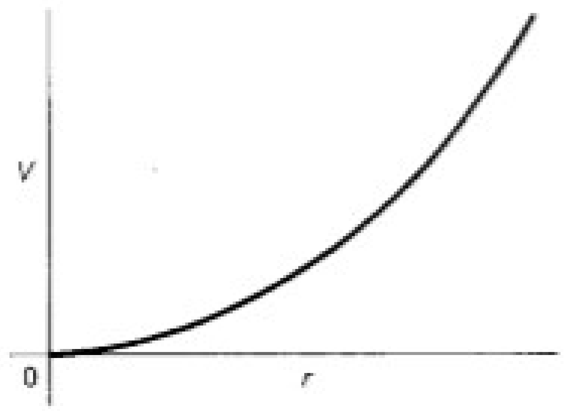


Fig. 2.1 The graph of $V = \frac{4}{3}\pi r^3$.

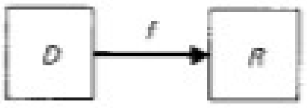
The function may not be defined for some values of the independent variable. For example, the volume of a sphere is not defined for negative values of the radius.

As an illustration of such a restriction in a commerical computation we might write the deduction due to personal exemptions in a tax collection as:

DEDUCTION = 750 * EXEMPTIONS;

It is usually the intent that this function DEDUCTION not be defined for cond tional or negative values of EXEMPTIONS, and later uses of DEDUCTION in the program may be based on the assumption that it is always a nonnegative integer. If, however, fractional exemptions are allowed for, say, the elderly or blind, then it must be a nonnegative integral multiple of some fixed fraction.

We call the set of values of the independent variable for which the function is defined the *domain* of definition of the function. The set of values that the function can assume is called its *range*. Every function then is a mapping from its domain of definition to its range.



The notion of a function can be extended in several ways. First, the
a) dependent variable may depend on several arguments, not just one. Perhaps this is a sophisticated concept, for the author recalls what in retrospect seems to have been interminable discussions in high school of whether human intelligence depended on heredity or environment. The idea that it might depend on both (even ignoring the relative weights of the two factors) seemed to be an elusive notion, difficult to comprehend. As a less controversial example, the pressure of a confined gas depends on both the volume and the temperature of the gas. Even if we do not know the exact nature of this dependence on two variables we can express the fact that such a dependence exists by writing

$$P = f(V, t),$$

indicating that the pressure P is some unspecified function of both the volume V and the temperature t.

In this case the functional correspondence is that between the pressure P and the *pair* of independent variables (V, t). To each pair of admissible values of V and t there corresponds some value of the pressure P. We might regard the pair of values (V, t) as a single object, or for that matter more generally regard an n-tuple (i.e., a list of n elements, sometimes called a "tuple" if n is unspecified) of arguments as a single object, and the domain as a collection of such objects. We can then still consider the function as associating some value in its range with each (single) object in its domain. The objects in the range can equally well be tuples of numbers rather than single values.

2.3 A FUNCTION CONSIDERED AS A MAPPING FROM ONE SET INTO ANOTHER

In all cases, then, a function is a mapping defined on the elements of a domain D and assuming as values the elements of a range R, and we might write for a function f with domain D and range R,

$$f: D \to R.$$

The important condition is that for each object x in the domain D there be one, and only one, object y in the range R. Sometimes y is called the "image" of x under f. The definition does permit two objects x_1, x_2 in D to have the same image y in R. Thus, if, we consider the correspondence that to each argument x in the domain of positive real numbers associates a positive *or* negative square root of x (that is, one or the other but not the pair or 2-tuple of square roots),

$$y = \pm\sqrt{x},$$

this is *not* an example of a function. However, usage in the literature has not always been consistent, and at times the phrase "single-valued function" has been used in the sense we have given for "function," with references to a "multivalued function" as something which violates the condition that the function's image in the range be unique for each element in the domain.

As noted earlier, if a function on the natural numbers is defined for all natural numbers it is said to be a *total* function; if it is defined for some, possibly not all, natural numbers it is a *partial* function. A total function, then, is also an example of a partial function (since "some" includes "all").

If the domain D of a function f is a subset of some set X (possibly all of X), we may say that f is partial on X. We can still write

$$\begin{tikzcd} \boxed{X} \arrow[r, "f"] & \boxed{R} \end{tikzcd}$$

or $f : X \to R$, even when f is not defined for all members of X.

We may also write

$$\begin{tikzcd} \boxed{X} \arrow[r, "f"] & \boxed{Y} \end{tikzcd}$$

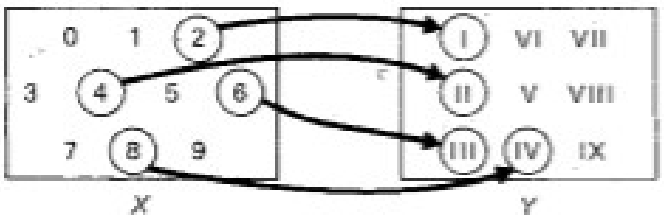
or $f : X \to Y$ even if the range of f is not all of Y. That is, f maps some of the elements of X into Y.

We say that the map f has the property of being "one-to-one" (sometimes called "injective") if to every element x in its domain D there corresponds a distinct element y in its range R. That is, no two elements in D are mapped into the same element in R. In this case we can invert the mapping, and we say that the "inverse" map from R to D, designated as "f^{-1}," exists. If $f(x) = y$, we can write $f^{-1}(y) = x$

where $f^{-1}: R \rightarrow D$. If $f: X \rightarrow Y$ is total on X, $f^{-1}: Y \rightarrow X$ is not necessarily total on Y.

If we have $f: X \rightarrow Y$, and the range of f is *all* of Y, i.e., the images of f "cover" Y, then the mapping f is said to be "onto" (sometimes called "surjective"), and f maps X "onto" Y. If f is a one-to-one onto mapping from X to Y, then f^{-1} exists and is total on Y.

Example. Consider the function $f: X \rightarrow Y$ where $X = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, $Y = \{\text{I, II, III, IV, V, VI, VII, VIII, IX}\}$, that maps a given decimal digit d into the Roman numeral representation of $d/2$.



We see that f is partial on X, into Y. The domain D of f is $\{2, 4, 6, 8\}$; the range R is $\{\text{I, II, III, IV}\}$. f is one-to-one, and f^{-1} exists. f^{-1} is partial on Y. Its domain is $\{\text{I, II, III, IV}\}$; its range is $\{2, 4, 6, 8\}$.

Example. Let X be the set of nonnegative integers. Consider the function $f: X \rightarrow X$ defined by $y = f(x) = x^2 - 3$. The domain D of f, or the set of values of x for which $f(x)$ is defined, is the set $\{2, 3, 4, 5, \ldots\}$. The range R of f, or the set of possible values of y, is the set $\{1, 6, 13, 22, \ldots\}$. $f(x)$ is one-to-one, for no two distinct numbers x_1, x_2 are mapped into the same number. The mapping f is not onto X since, for example, $f(x)$ never assumes the value 2.

The inverse map $f^{-1}: X \rightarrow X$ is defined by $x = f^{-1}(y) = \sqrt{y+3}$ (obtained by solving $y = x^2 - 3$ for x). The domain of f^{-1} is $\{1, 6, 13, 22, \ldots\}$, or, that is, it is the same as the range of f. The range of f^{-1} is $\{2, 3, 4, 5, \ldots\}$, the same as the domain of f.

2.4 ON SOME WAYS OF DEFINING FUNCTIONAL RELATIONSHIPS

There are several ways of *defining* functions mathematically. A function can be given *explicitly*, such as by the equation

$$Y = 3 * (X ** 3) - 2 * (X ** 2) + 5.$$

This equation describes Y as a function of X. A function may also be defined *implicitly*, such as by the equation

$$3 * X * (Y ** 2) - (X ** 2) * Y = 180$$

$3XY^2 - X^2Y = 180$

In this latter case, one would have to solve the given equation for a given value of X to find the corresponding value of Y. This corresponding Y would not in this instance be unique (which means Y would not be a function of X) unless we impose some further condition such as taking the larger (in absolute value) of the roots when there are two real roots.

In considering functions on the natural numbers, a definition may involve a *recursion* (cf. Chapter 4), where the defining expressions for the function may involve the function itself.

Consider, for example, the function defined by the two equations:

$$\begin{cases} f(0) = 1 \\ f(n) = n * f(n-1) \text{ for } n > 0. \end{cases}$$

These equations enable us to obtain $f(n)$ at $n = 0, 1, 2, \ldots$. We see that $f(1) = 1 * f(0) = 1, f(2) = 2 * f(1) = 2 * 1, f(3) = 3 * f(2) = 3 * 2 * 1, \ldots,$ and, in general, $f(n) = n * (n-1) * (n-2) * \ldots * 1 = n!$

We shall see later the key role that this technique for the definition of functions plays in computing. The example we have given involves, more precisely, the use of the operation called *primitive recursion* for defining a function $\phi(x)$. Two equations are given which enable us, first, to compute the function $\phi(x)$ at the starting value of zero and, second, to step along through the natural numbers, computing the function at any argument in terms of its value at the preceding argument. The notion is closely related to that of iteration, or the repeated application, of some procedure. We shall elaborate on these matters in Chapter 4.

Some programming languages include facilities for defining functions recursively. The following fragment of PL/1 code shows the above function defined recursively. Within the body of the procedure being defined, a reference is made to the procedure itself.

```
FACTORIAL: PROCEDURE (N) RECURSIVE RETURNS (FIXED);
           IF N = 0 THEN RETURN (1);
           ELSE RETURN (N * FACTORIAL (N - 1));
END FACTORIAL;
```

In computer applications, functions are frequently defined by storing tables. This is feasible if the domain includes only a finite, and not too large, set of values. In this case, the function $f(x)$ can be represented by storing in the machine, in some form, all pairs of values $(x, f(x))$ for all values of x that might appear in the course of the computation. (This set of pairs is called the *graph* of the function, cf. below.) A table lookup is required, rather than an arithmetic computation, to determine the value of $f(x)$ for a given x. Income tax tables for the lower range of incomes provide an obvious example of such functional representation.

Example. The graph of the function $f(x) = x^2 - 3$ is the set of pairs $\{(2, 1), (3, 6), (4, 13), (5, 22), \ldots\}$, where we consider f to be defined on the natural numbers.

The reader should recognize that our brief mention of these several ways to define functional relationships is given only to stress a few important notions; it is by no means complete. After all, every program provides a definition of a function, that which maps the input into the output, and in a superficial sense all of computer science is concerned with effective means to define functions.

2.5 FUNCTIONS DEFINED ON OBJECTS OTHER THAN THE NATURAL NUMBERS

In considering functions as mappings from one set into another there is no reason to restrict these sets to be sets of natural numbers, or even of n-tuples of natural numbers, and we have frequent occasion in computer science to deal with functions defined on other kinds of objects. We may, for example, consider collections of strings of symbols and mappings of these strings into other strings. For example, let Σ^* be the set of all finite strings formed from the symbols in the English alphabet (Σ is often used to designate the alphabet of some language). Consider the mapping HEAD that associates with each string its leading character, HEAD: $\Sigma^* \to \Sigma$. We have:

$$\text{HEAD(ABRACADABRA)} = \text{A}$$

The function, JOIN, defined on pairs of such strings is another example. This function maps each ordered pair (i.e., there is a *first* and a *second* member of each pair) of strings into the string formed by concatenating them.

$$\text{JOIN(CAR, FARE)} = \text{CARFARE}$$

We shall, in the sequel, use the words "map" or "mapping" and "function" interchangeably, and we shall consider mappings whose domains and ranges seem very different from the natural numbers. For example, we might take as the domain of a mapping a set of arithmetic statements about particular integers such as:

1. $2 + 3 = 4$	4. $6 < 4 - 5$
2. $7 < 10$	5. $6 \div 2 = 7 - 4$
3. $5 - 3 < 12$	6. $3 ** 3 = 9$

Each of these statements is either true (T or 1) or false (F or 0). We can then consider the mapping which associates to each such statement a value of 1 (or T) or 0 (or F). The given statements would then have the following values:

(1) 0, (2) 1, (3) 1, (4) 0, (5) 1, (6) 0.

In all cases the functions that arise in computer science involve domains and ranges that are *countable* sets (cf. the discussion in Section 3.7).

2.6 FUNCTIONS OF FUNCTIONS

We might even take *functions* themselves as objects of the domain of some mapping. Such a mapping might associate with each function in its domain some object such as a natural number or, possibly, some other function. We thus can consider a hierarchy of such mappings, starting with functions that map natural numbers into natural numbers

$$f : N \to N.$$

Then at a second level we would have mappings of functions into functions

$$F : f \to f$$

and perhaps, at a still higher level, mappings like

$$\phi : F \to F,$$

and so forth.

Examples of this sort of thing abound in mathematics and are fundamental to its development. A host of names including operator, transformation, morphism, functional, functor, etc., are used to refer to various situations of this kind.

In computing we can consider a program—a procedure—a routine—as providing a definition of some mapping of its input into its output. We may indicate symbolically that a particular program P designates such a mapping by writing

$$P : I \to O.$$

A compiler, then, would be an example in our hierarchy of a mapping of higher order, for it maps programs into programs

$$\text{compiler} : \text{programs} \to \text{programs}.$$

A "compiler-compiler" would be at a still higher, or a third, level, for it maps the specifications of a programming language into a compiler for that language.

Higher level mappings arise in the evolution of mathematics as we study sets, sets of sets, sets of sets of sets, and so forth. Perhaps comparable to this occurrence, an increasing sophistication is recognizable in some of the more advanced programming languages that provide features to deal with situations of this sort. PL/1 possesses this capability in a rudimentary way [3], ALGOL 68 has still more capability [6, 10], and SETL has considerable flexibility along these lines [8, 9]. For example, ALGOL 68 takes advantage of the fact that programs, routines, and procedures are themselves values and can therefore be parameters of other routines in calling statements.

Example. (ALGOL 68): An identity declaration like

$$\text{PROC(PROC)}\,f = (\text{PROC}\ a) : \text{E}$$

causes the calling statement

$$\text{CALL}\ f(x * y)$$

to result in the definition of a as a *procedure*. It is elaborated as

$$\text{PROC}\ a = x * y : \text{E}$$

Therefore, the value possessed by 'a' is a routine. Whenever the identifier 'a' appears in the elaboration of E, it results in this routine for the computation of the expression '$x * y$', and it is compiled in this way. Thus, the original procedure definition for f involves a higher level of definition. It does not define f as an ordinary function; it defines f as a function of a function. In our example, f associates the function $x * y$ with the function name a. Thus, f is a map from the domain of functions to the range of functions, $f : a \to x * y$.

The words "data type" are used with different meanings in the literature. The more common one, of course, refers to such entities as: integers, floating point numbers, fixed point numbers, complex numbers, alphabetic strings, bit strings, etc. However, some authors have defined data types of higher order to refer to the hierarchical situation we have described.

2.7 RELATIONS

If we abandon the requirement that a function associate a *unique* object in its range with every object in its domain, we are led to the notion of a relation. A function can be considered as being equivalent to its *graph*. Even when we cannot draw a picture such as Fig. 2.1, the graph of a function can still be regarded as the collection of all pairs (x, y) where x is an object in its domain and y is the associated object in its range (either or both objects may be n-tuples, $n > 1$, not just single numbers). From the requirement that there be a unique y for each x, it follows that if (x_1, y_1) and (x_1, y_2) are in the graph of a function f, then y_1 must necessarily be the same as y_2. This condition does not hold for a *relation*, which is defined to be *any* subset of pairs (x, y) without this restriction that no two different pairs have the same first component. The *domain* of a given relation is the set of all x's or first members of the pairs that constitute the relation; the *range* is the set of all y's or second members of these pairs.

We can describe a relation either by listing the pairs (x, y) that comprise it or, equivalently, by indicating that property which is satisfied by precisely these pairs. (See the definition of "predicate" in Section 2.12). The reasonableness of the name "relation" becomes apparent when we consider that "A is a brother of B" is an example of a relation. It involves as domain some human beings, and the pairs (A, B) that satisfy this stated relation are not restricted to having a unique B for given A. Examples of relations on the integers include: $x < y$, $x > y$, $x + 2 \neq y$. We note that an infinite number of pairs (x, y) with, say, $x = 7$ satisfy the relation $x < y$.

If the pair (x, y) is in the graph of a relation R we may write xRy to denote this fact. Equating the relation to its graph we can then say that "R is the set of pairs (x, y) such that xRy." This can be written, using conventional notation (cf. Section 3.4), as

$$R = \{(x, y) \mid xRy\}.$$

A relation R is said to be *reflexive* if for every element x in its domain we have xRx. It is *symmetric* if whenever xRy holds so does yRx. It is *transitive* if whenever we have both xRy and yRz it follows that xRz. The relation "A is a brother of B" is not transitive nor is it reflexive or symmetric (why not?).

A relation that possesses all three properties of being reflexive, symmetric, and transitive is called an *equivalence*, e.g., if $R = \{(x, y) \mid x - y \text{ is a multiple of } 7\}$, then R is an equivalence.

Note that the ordinary equality of numbers, $x = y$, is a relation that is reflexive ($x = x$), symmetric (if $x = y$ then $y = x$), and transitive (if $x = y$ and $y = z$, then $x = z$). Equality is thus an equivalence.

As another example, consider the following relation S defined on the domain consisting of all words in the English language. Let xSy if and only if the two words x, y both end in the same last three letters. It is quickly seen that S is an equivalence. Other examples appear in the exercises.