CHAPTER 1

Discrete Systems

Discrete mathematics is the study of techniques, ideas and modes of reasoning that are indispensable in applied disciplines such as computer science or information technology. It is also a gateway into advanced theoretical mathematics.

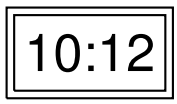
To understand its focus, it is helpful to appreciate the meaning of—and differences between—analog and discrete systems. Discrete mathematics is the study of *discrete* (as opposed to *analog*) systems.

1.1 Analog versus Discrete Systems

The difference between analog and discrete systems is that analog systems involve smooth, continuous, unbroken movement or structures, whereas discrete systems involve individual parts or states that are clearly separate from one another.

This is illustrated by the difference between a traditional (analog) clock and a digital (discrete) clock. The hands of an analog clock move in a fluid, continuous motion. In the one minute between 10:12 and 10:13, the minute hand moves in a smooth, unbroken motion passing through all instants between these two times. In an hour it passes through infinitely many different instants of time. This is an analog system. By contrast, a digital clock jumps from 10:12 to 10:13 in an instant. In an hour it records a finite (in fact, 60) instants of time. This is a *discrete* system.

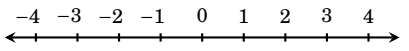
Analog clock



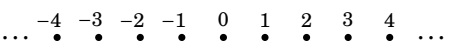
Discrete (digital) clock

Although calculus is not really essential to mastering most topics in discrete mathematics, it can help us gain a deeper understanding the difference between analog and discrete systems. Calculus is based on the

real number system, which is an analog system. We visualize the real numbers as a smooth, unbroken, infinitely long line. You can put your finger on 0 and move it continuously to the right in a fluid motion, stopping at (say) 3. As you do this, your finger moves through infinitely many numbers, one for each point on the line from 0 to 3. This is an analog system.



Real numbers (analog system)



Integers (discrete system)

Discrete mathematics is more concerned with number systems such as the integers (whole numbers) $\ldots -3, -2, -1, 0, 1, 2, 3 \ldots$ whose parts (numbers in this case) are discrete entities. Putting your finger at 0 and moving to the right, you jump from 0, to 1, to 2, to 3, and so on.

But discrete mathematics deals with much more than just integers. More broadly, it encompasses mathematical structures or processes that consists of individual parts. You can think of discrete mathematics as the discipline concerned with mathematical structures whose parts can be described by a finite sequence of characters from a computer keyboard.

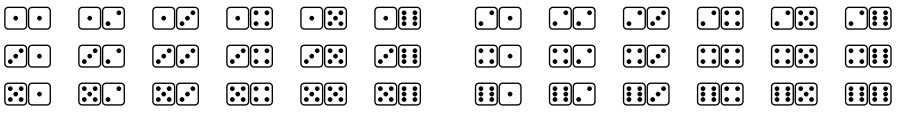
For example, the set of integers is a discrete mathematical system because even though there are infinitely many integers (and you'd never be finished typing all of them), any one integer can be expressed by typing a finite sequence of the digits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, and possibly a minus sign "–" (for negative integers).

Similarly, the set of rational numbers (fractions of integers) is also a discrete system because any rational number, such as −397/24 is expressible as a finite sequence of the symbols $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, -$ and /.

By contrast, the system of real numbers is **not** a discrete system: it has irrational numbers like $\pi = 3.14159265359\ldots$ that cannot be typed because they involve infinitely many decimal places.

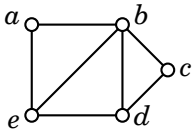
In this sense, the times represented by a digital clock are a discrete system because any time can be expressed with an integer from 1 to 12, followed by a colon, and then a number from 00 to 59. (As in 5:00, or 11:45.) An analog clock is inherently different. For an instant, an analog clock reads π o'clock, which occurs a small fraction of a second after 3:14. But a digital clock jumps from 3:14 to 3:15, blissfully ignoring the messiness of irrational numbers such as π. This is the spirit of discrete mathematics.

Even the process of rolling a dice two times in a row is a discrete system, as there are exactly 36 outcomes.



Any one of these outcomes could be encoded in symbols, as (x,y), where x and y are integers between 1 and 6, representing the results of the first and second rolls. (For example,  is encoded as $(3,5)$, etc.) The set of all such outcomes is called a *sample space*. Such sets can be useful. For example, if we wanted to know the probability of rolling a double, we can see that exactly 6 of these 36 equally likely outcomes is a double, so the probability of rolling a double is $\frac{6}{36} = \frac{1}{6}$. Chapters 2 and 4 introduce sets and counting, theories relevant to situations such as this, and we investigate the theory of probability in Chapter 5.

Another example of a discrete structure is a *graph*. In mathematics, the word "graph" is used in two different contexts. In algebra or calculus a *graph* is a visual description of a function, graphed on a coordinate axis. Although we do use such graphs in discrete mathematics, we more often use the word a *graph* to mean a network of nodes with connections between them. Here is a picture of a typical graph.



Its nodes are described by the discrete set $\{a,b,c,d,e\}$, and its connections (called *edges*) are $\{ab,bc,cc,de,ea,eb\}$. Therefore this particular graph is completely described by typing the information

$$\Big(\{a,b,c,d,e\},\quad \{ab,bc,cc,de,ea,eb\}\Big).$$

The theory of graphs is a major branch of discrete mathematics. Graphs have wide ranging applications. For example, the Internet is a huge graph whose nodes are web pages and whose edges are links between them. Google's search algorithm involves the mathematics of this structure.

The fact that a graph can be described by sets of vertices and edges is another indication of the fundamental importance of sets, which we will study carefully in Chapter 2.

This book is organized as follows. Chapter 2 introduces the theory of sets, a language capable of describing any discrete (or analog) structure. This is followed by a short chapter on logic, a system that bridges the gap between natural language and mathematics, giving precision to the modes of speech we use in discussing mathematics.

Chapters 4 and 5 build on this, laying the foundation for enumerating or *counting* the parts of discrete structures. Chapter 6 introduces the study of *algorithms*. a fundamental idea that is at the heart of computer science. Chapter 7 covers some additional topics in logic.

As you delve deeper into mathematics and its applications, you will find yourself in situations where you need to *prove* that a certain result is correct (i.e., true), and you will need to read and understand proofs written by others. This is the topic of chapters 8 through 14. This is a major part of the text, and it will build on the previous chapters. This material is applied in Chapter 15, a brief introduction to graph theory. Chapters 16 and 17 deal with the important topics of relations and functions. All of this material is essential for real progress in mathematics and computer science.

Although calculus is not necessary to understand the ideas in this book, you have probably studied it, and that background will serve you well. For instance, calculus requires a certain fluency in algebra and arithmetic, and that fluency is equally essential in discrete mathematics. Calculus requires a working knowledge of functions, and that background will be useful. It has also given you a grounding in certain useful notations, such as the sigma notation for expressing sums. Given a list of numbers $a_1, a_2, a_3, \ldots, a_n$, their sum is compactly expressed as

$$a_1 + a_2 + a_3 + \cdots + a_n = \sum_{k=1}^{n} a_k.$$

All of these background topics will play a role for us.

Before beginning with the theory of sets, we pause to review the binary and hexadecimal number systems. Although not absolutely fundamental for most of this text, they are important because they are the number systems that form the basis for the internal workings of computer circuitry and computations. These systems will also show up in certain examples and exercises throughout the text.

Exponential notation makes an appearance here. Recall that for any number a and positive integer n, the power $a^n = a \cdot a \cdot \cdots \cdot a$ is the product of a with itself n times. Recall from algebra that if a is non-zero, then $a^0 = 1$. In the following pages you will encounter $10^0 = 1$, $2^0 = 1$ and $16^0 = 1$.

1.2 The Binary Number System

In daily life we use the familiar base-10 number system, also called the **Hindu-Arabic** number system, or the **decimal** number system. It uses ten symbols $0,1,2,3,4,5,6,7,8,9$, representing the quantities zero through nine. There is no single symbol for the quantity ten – instead we express it as the combination "10," signifying that ten equals 1 ten plus 0 ones. Any other positive integer is represented as a string of symbols $0,1,2,3,4,5,6,7,8,9$, standing for the sum of the digits in the string times powers of ten, decreasing to the zeroth power $10^0 = 1$. For example,

$$\begin{aligned} 7406 &= \mathbf{7}\cdot 10^3 + \mathbf{4}\cdot 10^2 + \mathbf{0}\cdot 10^1 + \mathbf{6}\cdot 10^0 \\ &= \mathbf{7}\cdot 1000 + \mathbf{4}\cdot 100 + \mathbf{0}\cdot 10 + \mathbf{6}\cdot 1. \end{aligned}$$

Thus the number seven-thousand-four-hundred-six is represented as 7406, with a 7 in the *thousand's place*, a 4 in the *hundred's place*, a 0 in the *ten's place*, and a 6 in the *one's place*. There is little need to elaborate because you internalized this early in life.

There is nothing sacred about base of ten, other than the fact that it caters to humans (who have ten fingers). If we had eight fingers, our number system would surely be base-8. Actually, for any integer $n > 1$ there is a base-n number system using n symbols. Although base-10 is convenient for humans, base-2 is better suited for computer circuitry, because its two symbols can be represented by a zero or positive voltage.

The **base-2**, or **binary** number system uses only two digits, 0 and 1, representing the quantities zero and one. There is no single symbol for the number two – instead we express it as the combination "10," signifying that two equals 1 two plus 0 ones. Any other quantity is represented as a string of the symbols $0,1$, standing for the sum of the digits in the string times powers of two, decreasing to $2^0 = 1$.

For example, the base-2 number **10011** equals the base-10 number

$$\begin{aligned} &\mathbf{1}\cdot 2^4 + \mathbf{0}\cdot 2^3 + \mathbf{0}\cdot 2^2 + \mathbf{1}\cdot 2^1 + \mathbf{1}\cdot 2^0 \quad = \\ &\mathbf{1}\cdot 16 + \mathbf{0}\cdot 8 + \mathbf{0}\cdot 4 + \mathbf{1}\cdot 2 + \mathbf{1}\cdot 1 \quad = \quad 19. \end{aligned}$$

The number nineteen is represented as "10011" in base-2 because it is the sum of **1** sixteen, **0** eights, **0** fours, **1** two and **1** one. It is represented as "19" in base-10 because it is the sum of **1** ten and **9** ones.

For clarity, we sometimes use a subscript to indicate what base is being used, so the above computation is summarized as $10011_2 = 19_{10}$.

Table 1.1 shows the first sixteen decimal numbers in the left column, with their corresponding binary representations on the right. Be sure you agree with this. For instance, $110_2 = 6_{10}$ because $\mathbf{1}\cdot 2^2 + \mathbf{1}\cdot 2^1 + \mathbf{0}\cdot 2^0 = 6$.

binary number		powers of 2 16 8 4 2 1		decimal number
0	=	**0**·1	=	0
1	=	**1**·1	=	1
10	=	**1**·2+**0**·1	=	2
11	=	**1**·2+**1**·1	=	3
100	=	**1**·4+**0**·2+**0**·1	=	4
101	=	**1**·4+**0**·2+**1**·1	=	5
110	=	**1**·4+**1**·2+**0**·1	=	6
111	=	**1**·4+**1**·2+**1**·1	=	7
1000	=	**1**·8+**0**·4+**0**·2+**0**·1	=	8
1001	=	**1**·8+**0**·4+**0**·2+**1**·1	=	9
1010	=	**1**·8+**0**·4+**1**·2+**0**·1	=	10
1011	=	**1**·8+**0**·4+**1**·2+**1**·1	=	11
1100	=	**1**·8+**1**·4+**0**·2+**0**·1	=	12
1101	=	**1**·8+**1**·4+**0**·2+**1**·1	=	13
1110	=	**1**·8+**1**·4+**1**·2+**0**·1	=	14
1111	=	**1**·8+**1**·4+**1**·2+**1**·1	=	15
10000	=	**1**·16+**0**·8+**0**·4+**0**·2+**0**·1	=	16

Table 1.1. Binary and decimal representations of numbers

In converting between binary and decimal representations of numbers, it's helpful to know the various powers of 2. They are listed in Table 1.2. For example, $2^5 = 2\cdot 2\cdot 2\cdot 2\cdot 2 = 32$, so 32 appears below 2^5.

…	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
…	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

Table 1.2. Powers of 2

Table 1.1 suggests a method for converting binary numbers to decimal. To convert a given a binary number to decimal, multiply its digits by decreasing powers of two, down to $2^0 = 1$, and add them. For example,

$$\begin{aligned} 1110 &= \mathbf{1}\cdot 2^3 + \mathbf{1}\cdot 2^2 + \mathbf{1}\cdot 2^1 + \mathbf{0}\cdot 2^0 \\ &= \mathbf{1}\cdot 8 \;+\; \mathbf{1}\cdot 4 \;+\; \mathbf{1}\cdot 2 \;+\; \mathbf{0}\cdot 1 \quad = \quad 14. \end{aligned}$$

Example 1.1 Convert the binary number 110101 to decimal.

Solution: We simply write this number as a sum of powers of 2 in base-10.

$$\begin{aligned} 110101 &= \mathbf{1}\cdot 2^5 + \mathbf{1}\cdot 2^4 + \mathbf{0}\cdot 2^3 + \mathbf{1}\cdot 2^2 + \mathbf{0}\cdot 2^1 + \mathbf{1}\cdot 2^0 \\ &= \mathbf{1}\cdot 32 + \mathbf{1}\cdot 16 + \mathbf{0}\cdot 8 + \mathbf{1}\cdot 4 + \mathbf{0}\cdot 2 + \mathbf{1}\cdot 1 \quad = \quad 53 \end{aligned}$$

Thus $110101_2 = 53_{10}$, that is, 110101 (base 2) is 53 (base 10). ✍

Converting decimal to binary involves running this process in reverse, which can involve some reverse engineering.

Example 1.2 Convert the decimal number 347 to binary.

Solution: We need to find how 347 is a sum of powers of 2. Table 1.2 shows that the highest power of 2 less than 347 is $2^8 = 256$, and

$$\begin{aligned} 347 &= 256 + 91 \\ &= 2^8 + 91. \end{aligned}$$

Now look at the 91. Table 1.2 shows that the highest power of 2 less than 91 is $2^6 = 64$, and $91 = 64 + 27 = 2^6 + 27$, so the above becomes

$$347 = 2^8 + 2^6 + 27.$$

From here we can reason out $27 = 16 + 8 + 2 + 1 = 2^4 + 2^3 + 2^1 + 2^0$. Therefore

$$347 = 2^8 + 2^6 + 2^4 + 2^3 + 2^1 + 2^0.$$

Powers 2^7, 2^5 and 2^2 do not appear, so we insert them, multiplied by 0:

$$347 = \mathbf{1}\cdot 2^8 + \mathbf{0}\cdot 2^7 + \mathbf{1}\cdot 2^6 + \mathbf{0}\cdot 2^5 + \mathbf{1}\cdot 2^4 + \mathbf{1}\cdot 2^3 + \mathbf{0}\cdot 2^2 + \mathbf{1}\cdot 2^1 + \mathbf{1}\cdot 2^0.$$

Therefore 347 is the base-2 number 101011011. ✍

Various cultures throughout history have used base-n number systems. The ancient Babylonians used a base-60 system with 60 different cuneiform digits (including a blank, used for what we now call 0). The Aztecs used base-20. In the modern era, some early computers used the base-3 system, with three digits represented by a positive, zero or negative voltage.

Today the binary system is the foundation for computer circuitry, with 0 represented by a zero voltage, and 1 by a positive voltage. Though the binary system has just two digits, it is inefficient in the sense that many digits are needed to express even relatively small numbers. Base-16 remedies this. It is closely related to binary, but it is much more compact.

1.3 The Hexadecimal Number System

Base-16 is called the **hexadecimal** number system. It uses 16 symbols, including the familiar ten symbols $0,1,2,3,4,5,6,7,8,9$ representing the numbers zero through nine, plus the six additional symbols A, B, C, D, E, and F, representing the numbers ten through fifteen.

Table 1.3 summarizes this. It shows the numbers zero through fifteen in decimal, binary and hexadecimal notation. For consistency we have represented all binary numbers as 4-digit strings of 0's and 1's by adding zeros to the left, where needed.

decimal	binary	hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Table 1.3. The first sixteen integers in decimal, binary and hexadecimal

The number sixteen is represented as 10 in hexadecimal, because sixteen is **1** sixteen and **0** ones. Note $16_{10} = 10_{16} = 10000_2$.

Just as powers of two are fundamental to interpreting binary numbers, powers of sixteen are necessary for understanding hexadecimal. Here are the first few powers. (Memorizing these is *not* essential.)

$\cdots$	16^5	16^4	16^3	16^2	16^1	16^0
$\cdots$	1,048,576	65,536	4096	256	16	1

Table 1.4. Powers of 16

We can convert between hexadecimal and decimal in the same way that we converted between binary and decimal.

Example 1.3 Convert the hexadecimal number 1A2C to decimal.

Solution: Simply write 1A2C as a sum of powers of sixteen in hexadecimal, then convert the sums to decimal. (In interpreting the first line, recall that 10 is the hexadecimal representation of sixteen, i.e., $10_{16} = 16_{10}$.)

$$\begin{aligned} 1A2C &= \mathbf{1}\cdot 10^3 + \mathbf{A}\cdot 10^2 + \mathbf{2}\cdot 10^1 + \mathbf{C}\cdot 10^0 && \text{(hexadecimal)} \\ &= \mathbf{1}\cdot 16^3 + \mathbf{10}\cdot 16^2 + \mathbf{2}\cdot 16^1 + \mathbf{12}\cdot 16^0 && \text{(decimal)} \\ &= \mathbf{1}\cdot 4096 + \mathbf{10}\cdot 256 + \mathbf{2}\cdot 16 + \mathbf{12}\cdot 1 && \text{(decimal)} \\ &= 6700 && \text{(decimal)} \end{aligned}$$

Thus $1A2C_{16} = 6700_{10}$, that is, 1A2C (base 16) is 6700 (base 10). ✍

Converting between hexadecimal and binary is extremely simple. We will illustrate the technique first, before explaining why it works. Suppose we wish to convert the binary number 1111 1100 1000 0010 11 to hexadecimal. The first step is to divide the digits of this binary number into groups of four, beginning from the right.

$$11 \quad 1111 \quad 0010 \quad 0000 \quad 1011$$

If necessary, add extra zeros to left end of the left-most grouping, so that it too contains four digits.

$$0011 \quad 1111 \quad 0010 \quad 0000 \quad 1011$$

Now use Table 1.3 (or innate numerical reasoning) to convert each 4-digit binary grouping to the corresponding hexadecimal digit.

$$\begin{array}{ccccc} 0011 & 1111 & 0010 & 0000 & 1011 \\ 3 & F & 2 & 0 & B \end{array}$$

We conclude that $1111\,1100\,1000\,0010\,11_2 = 3F20B_{16}$.

The reverse process works for converting hexadecimal to binary. Suppose we wanted to convert 1A2C to binary. Taking the reverse of the above approach (and using Table 1.3 if necessary), we write

$$\begin{array}{cccc} 1 & A & 2 & C \\ 0001 & 1010 & 0010 & 1100. \end{array}$$

Ignoring the three 0's on the far left, we see $1A2C_{16} = 1\,1010\,0010\,1100_2$.

It is easy to see why this technique works. Just use the computation from Exercise 1.3 on page 11, but convert 1A2C to binary instead of decimal. (Here we use the fact that $10_{16} = 10000_2$.)

$$\begin{aligned} 1A2C &= \mathbf{1}\cdot 10^3 + \mathbf{A}\cdot 10^2 + \mathbf{2}\cdot 10^1 + \mathbf{C}\cdot 10^0 && \text{(base-16)} \\ &= \mathbf{1}\cdot 10000^3 + \mathbf{1010}\cdot 10000^2 + \mathbf{10}\cdot 10000^1 + \mathbf{1100}\cdot 10000^0 && \text{(binary).} \end{aligned}$$

Doing the addition in columns, we get:

$$\begin{array}{rr} & \mathbf{1}\ 0000\ 0000\ 0000 \\ & \mathbf{1010}\ 0000\ 0000 \\ & \mathbf{10}\ 0000 \\ + & \mathbf{1100} \\ \hline & 1\ 1010\ 0010\ 1100 \end{array}$$

This is the same number we would get by replacing each digit in 1A2C with its binary equivalent.

Exercises for Chapter 1

A. Convert the decimal number to binary and hexadecimal.

1. 347	**2.** 10,000	**3.** 2039	**4.** 64
5. 256	**6.** 257	**7.** 258	**8.** 258

B. Convert the binary number to hexadecimal and decimal.

9. 110110011	**10.** 10101010	**11.** 1111111	**12.** 111000111
13. 101101001	**14.** 10011010	**15.** 1000001	**16.** 100100101

C. Convert the hexadecimal number to decimal and binary

17. 123	**18.** ABC	**19.** 5A4D	**20.** F12
21. B0CA	**22.** C0FFEE	**23.** BEEF	**24.** ABBA
