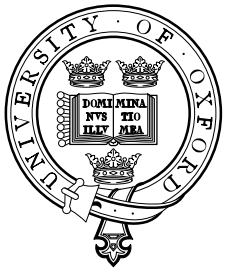
# Discrete Mathematics

**Andrew D. Ker**

16 Lectures, Michaelmas Term 2010




Oxford University Computing Laboratory

# Contents

# Introduction To The Lecture Notes

## Course

The **Discrete Mathematics** course is compulsory for first-year undergraduates in Computer Science. There are 16 lectures, supported by tutorials arranged by college tutors.

## Prerequisites

None.

## Syllabus

Sets: union, intersection, difference, power set, algebraic laws. Functions: injectivity & surjectivity, composition & inverse. Relations, equivalence relations, and partitions; relational composition & converse, transitive closure; orders, least upper and greatest lower bounds. Combinatorial algebra: permutations, combinations, sums of finite sequences. Functions associated with combinatorial problems: ceiling, floor, factorial, combinatorial coefficients. The inclusion-exclusion principle. Recurrence relations arising from combinatorial problems. Modular arithmetic, Euclid's algorithm, and applications.

## Outline of Lectures

The material is presented in eight sections, one per week. Each section is concerned with one topic in discrete maths, and a technique for constructing mathematical proofs.

Sets: Definition of sets, subsets, some standard sets; union, intersection, relative complement, symmetric difference, complement, cartesian products, power sets; algebraic laws; cardinality of finite sets. Writing mathematical proofs, double inclusion proofs for set equality, proof by cases. Extra topic: bags.

Functions: Intervals; functions, domain and codomain, partial functions, restriction; 1-1, onto, and bijective functions; composition and inverse. Proof of the contrapositive and proof by contradiction. Extra topic: binary operators.

Counting: Counting laws of sum, subtract, product; permutations, factorial function, combinations, binomial coefficients; inclusion-exclusion principle, derangements; floor and ceiling functions. Proofs techniques for counting, with many examples. Extra topic: multinomial coefficients.

Relations: Binary relations; properties of relations: reflexivity, irreflexivity, symmetry, antisymmetry, transitivity, seriality; equivalence relations, equivalence classes, and partitions; relational composition, converse, and transitive closure; graphical representation of relations. Extra topic: counting relations.

Sequences: Sequences and recurrence relations; sigma notation and partial sums of sequences; recurrence relations arising from counting problems, including derangements and partitions. Proof by induction. Extra topic: solving linear recurrence relations.

Modular Arithmetic: Definition of modular arithmetic via an equivalence relation; properties of addition, multiplication, and exponentation $(\mathrm{mod}\ n)$; Euclid's algorithm, binary MOD and DIV functions, multiplicative inverses $(\mathrm{mod}\ p)$. The Pigeonhole Principle and many examples. Extra topic: modular square roots of $-1$.

Asymptotic Notation: Big-O notation, tail behaviour of sequences, examples drawn from running times of common algorithms. Proofs of sentences of the form $\exists x.\forall y.P$, with examples of asymptotic behaviour proofs including simplified Stirling's formula. Choosing the right inductive hypothesis for big-O proofs. Extra topic: solving recurrence relations of divide-and-conquer type, up to asymptotic order.

Orders: Pre-orders, partial orders, and linear orders; chains; product and lexicographic order on cartesian products; upper and lower bounds, lub and glb. Proofs of sentences of the form $\forall x.\exists y.P$, with examples from interesting orders. Extra topic: order isomorphisms.

## Reading Material

The lecture notes provide all the necessary definitions, some examples, and some exercises. It would be a good idea, though, to buy or borrow a textbook in order to supplement and round out the material (and particularly for finding more practice exercises). The recommended text is

**K. A. Ross and C. R. B. Wright, Discrete Mathematics (Fifth Edition), Prentice Hall, 2003.**

This book has much to commend it, including an enormous number of examples and exercises and a computer science oriented exposition. It is rather expensive (about £50) but there are many copies in Oxford libraries.

Three other books, which are fairly useful, are:

**R. P. Grimaldi, Discrete And Combinatorial Mathematics (Fifth Edition), Addison Wesley, 2003.**

A bit strange, with some very advanced topics alongside the standard material. Its big advantage is comprehensive coverage of the course material and lots of good practice exercises (some difficult). Probably not worth buying, but a good book to borrow. The fifth edition is the most recent but earlier editions are equally good.

**A. Chetwynd and P. Diggle, Discrete Mathematics, Arnold, 1995.**

The complete opposite, this book covers the basics well. It is easy to read and worth consulting if you are struggling, but only covers the first half of the material in this course. Cheap.

**P. Grossman, Discrete Mathematics for Computing (Third Edition), Palgrave MacMillan, 1995.**

Moderate level with an emphasis on modern computer science applications. Covers about three quarters of our material and includes other useful chapters on logic (will appear in the Digital Hardware course) and graphs (will appear in Design and Analysis of Algorithms). Cheap.

## Practice Questions and Tutorial Exercises

At the end of each chapter are some **practice questions**: usually short and fairly simple, they are to help you test your own understanding. Brief answers are provided on the final page of the chapter. It is recommended to try the practice questions soon after the relevant lectures, to help you correct any misunderstanding quickly.

More substantial questions are set in the **tutorial exercises**, provided separately from the notes. There will be **four** regular tutorial sheets, plus one additional sheet covering the last week's lecture material along with general revision suitable for vacation work (whether to set the vacation work, or indeed whether to follow these exercises at all, is a decision for college tutors). Model answers are provided for tutors' use only.

It is not expected that your tutors will want to discuss the practice questions and so it is possible to get by without doing them. However, they have been designed so that some of the tutorial exercises will be easier if the practice questions have been attempted beforehand.

## Course Website

Course material can be found at `http://web.comlab.ox.ac.uk/teaching/courses/discretemaths/`. The lecture notes will be published about a week **after** the corresponding lectures (if you want timely notes, you will have to attend), and the tutorial exercise sheets will appear in weeks 1, 3, 5, 7 (and 8 for the vacation work).

# Chapter 1

# Sets

---

**Reading:** Ross & Wright: 1.3, 1.4;
Chetwynd & Diggle: 2.1–2.6;
Grimaldi: 3.1, 3.2;
Grossman: 5.1–5.3.

---

It is necessary to understand the concept of sets to express anything, be it mathematics or the behaviour of computer systems, formally. Furthermore, the notation of sets provides a concise way to express statements about computers. Much of the material in this chapter is probably familiar, but it is important to begin thoroughly, and sets are also a useful topic with which to study the writing of mathematical proofs, something which will shadow the discrete mathematics material throughout this course.

## 1.1 Defining Sets

Although the idea of a set as a collection of objects (in which neither order nor duplication are significant) is simple enough, it is rather complicated to give a *formal* definition from scratch. We will leave the complexities to mathematical philosophers and adopt a simple working definition, but you should know that the subject called *axiomatic set theory* is deep and difficult.

**Definition** A **set** is an unordered collection of distinct objects. When $x$

is in the set $A$ we say that $x$ **is an element (or member) of** $A$ and write $x \in A$. When $x$ is not an element of $A$ we write $x \notin A$.

A set can be defined by its members, because two sets are called equal (written $A = B$) if they have the same members. We can therefore specify a set simply by listing its members, for example $A = \{1, 2, 3\}$ which means the set $A$ which has members 1, 2, 3. We can say that $1 \in A$ and $4 \notin A$. Notice the "curly brackets" (called **braces**) which are traditionally used to define sets. Because neither duplicates nor order are significant, $\{3, 2, 1, 2\}$ represents the same set.

Sets can have infinitely many members, and it is not necessary to list them all explicitly as long it is clear what is and what is not a member. For example the description $\{1, 3, 5, 7, \ldots\}$ unambiguously defines the positive odd integers. Another way to define a set is by a **set comprehension**:

$$\{x \mid \text{some condition on } x\}$$

which denotes the set whose members are all those meeting the condition. For example,

$$\{x \mid x \in \{1, 2, 3, 4, 5, 6, 7, 8, 9\} \text{ and } x \text{ is divisible by } 4\}$$

is an over-complicated definition of the set $\{4, 8\}$. The symbol "|" is sometimes alternatively written as ":"; it is pronounced "such that". (A note for the formalist: axiomatic set theory has particular things to say about set comprehensions – it is possible to misuse them to define things which are not actually sets – but that will never be an issue for us in this course.)

There are two shorthands often used with set comprehensions. First, if the condition includes membership of a set, that part of the condition is sometimes written before the bar, for example

$$E = \{n \in \mathbb{N} \mid n/2 \in \mathbb{N}\}$$

as a definition of the even integers. Second, one sometimes writes a function before the bar (functions are the subject of Chapter 2): all values of the function, whose arguments satisfy the condition, define the set. In this way the set $E$ can be alternatively defined by

$$E = \{2m \mid m \in \mathbb{N}\}.$$

A note on alphabets. There is no fundamental distinction between "element" and "set"; it is entirely possible for one set to be a member of another, for example $\{\{1,2\},\{2,3\}\}$ is a set with two members $\{1,2\}$ and $\{2,3\}$, each of which is a set of numbers. (However, it is usually disallowed for any set to be a member of itself; this is another philosophical question we intend to avoid.) Many books adopt the practice of using lowercase Roman letters like $x$ and $y$ for members of sets, and uppercase letters like $A$ and $B$ for sets; then they sometimes use calligraphic letters like $\mathcal{A}$ and $\mathcal{B}$ for sets of sets. However this distinction is artificial (take for example the set $\{1,\{1\}\}$: it has one member which is a number, and another which is a set) and one quickly runs out of different alphabets if there are sets of sets of sets of...

**Definition** Some standard sets are

| | | |
|---|---|---|
| (i) | $\emptyset = \{\}$, | the **empty set**. |
| (ii) | $\mathbb{N} = \{0,1,2,\ldots\}$, | the **natural numbers** (or **nonnegative integers**). |
| (iii) | $\mathbb{N}_+ = \{1,2,3,\ldots\}$, | the **positive integers**. |
| (iv) | $\mathbb{Z} = \{\ldots,-2,-1,0,1,2,\ldots\}$, | the **integers**. |
| (v) | $\mathbb{Z}_n = \{0,1,2,\ldots,n-1\}$, | the **integers modulo** $n$ (for $n = 2,3,4,\ldots$). |
| (vi) | $\mathbb{Q}, = \left\{\frac{n}{d} \;\middle|\; n \in \mathbb{Z} \text{ and } d \in \mathbb{N}_+\right\}$, | the **rational numbers**. |
| (vii) | $\mathbb{R}$, | the **real numbers**. |

It is unambiguous to write $\mathbb{Z}_+$ instead of $\mathbb{N}_+$, but beware! Some people write $\mathbb{N}$ for the set of positive integers and something like $\mathbb{N}_0$ for what we have called $\mathbb{N}$. And some write $\mathbb{P}$ for the set of positive integers, while others use the same symbol for the set of prime numbers. It is truly unfortunate that there is no agreement on such basic terminology. The definitions we use are probably the most common but it is important to check which version is being used whenever you consult a textbook.

Finally, note the distinction between $\{\}$ and $\{\{\}\}$; the first is an empty set with no elements, whereas the second is a set with one element (and this element happens to be another set).

## 1.2 Comparing Sets and Writing Proofs

There are ways to compare sets. We have already seen the first of the following:

**Definition** When sets $A$ and $B$ have exactly the same members we write $A = B$.

When all the elements of $A$ are also members of $B$ we say that $A$ is a **subset** of $B$, and write $A \subseteq B$. Two alternative ways of saying the same thing are that $A$ is **included** in $B$ or that $B$ is a **superset** of $A$, written $B \supseteq A$.

When $A \subseteq B$ and $A \neq B$ we say that $A$ is a **proper subset** of $B$, written $A \subset B$ or $B \supset A$. (This means that all members of $A$ are also members of $B$, and further that some members of $B$ are not members of $A$.)

Beware! Some people write $A \subset B$ to mean that $A$ is a subset of $B$, and use the symbol $A \subsetneq B$ or $A \subsetneqq B$ for a proper subset.

For example, we have $\{2, 3\} \subset \{1, 2, 3\}$, $\mathbb{N}_+ \subset \mathbb{N} \subset \mathbb{Z}$, and both $\emptyset \subseteq A$ and $A \subseteq A$ for *any* set $A$.

These definitions are also important for what they tell us about *proof*. Methods for constructing proofs are an important part of this course. First, what is a proof? Again this can be cast as a philosophical question, and for now we will adopt a simplified definition. Whether implicitly or explicitly, practically all proofs are of statements of the form "if (some hypotheses) then (a conclusion)". A **proof** is a sequence of statements which begin with the hypotheses and end with the conclusion, and where each step in the sequence follows logically from (some of) the previous steps.

The preceding definitions tell us what the form of proofs about sets should be. If we want to prove "if (some hypotheses) then $A \subseteq B$" then we should begin by assuming the hypotheses, suppose that $x \in A$ for an arbitrary $x$, and try to deduce that $x \in B$. If we want to prove "if (some hypotheses) then $A = B$" then it is usual to prove both $A \subseteq B$ and $A \supseteq B$. This is called a **double inclusion proof**. Sometimes the two halves of a double inclusion proof can be done in a single, reversible, proof. We will see examples of these mathematical proofs about sets in the next section.

## 1.3 Unions, Intersections, and Algebraic Laws

Now we introduce the most basic operations for combining sets.

**Definition** The **union** of sets $A$ and $B$, written $A \cup B$, is the set whose elements are in $A$ or in $B$ (or both, which normally goes without saying):

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

The **intersection** of sets $A$ and $B$, written $A \cap B$, is the set whose elements are in both $A$ and $B$:

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

When $A \cap B = \emptyset$ we say that $A$ and $B$ are **disjoint**.

For example, if $A = \{0, 1, 2, 3\}$ and $E = \{n \in \mathbb{N} \mid n \text{ is even}\}$ then $A \cup E = \{0, 1, 2, 3, 4, 6, 8, \ldots\}$ and $A \cap E = \{0, 2\}$. If $O = \{n \in \mathbb{N} \mid n \text{ is odd}\}$ then $E \cup O = \mathbb{N}$ and $E \cap O = \emptyset$ ($E$ and $O$ are disjoint).

There are many equations involving $\cup$ and $\cap$, which hold for all sets. Such equations are called **algebraic laws**.

**Claim 1.1** For any sets $A$, $B$ and $C$, the following are true.

The **idempotence** laws for $\cup$ and $\cap$:

$A \cup A = A$, $\qquad A \cap A = A$.

The **commutative** laws:

$A \cup B = B \cup A$, $\qquad A \cap B = B \cap A$.

The **associative** laws:

$(A \cup B) \cup C = A \cup (B \cup C)$, $\qquad (A \cap B) \cap C = A \cap (B \cap C)$.

The **distributive** laws:

$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$, $\qquad A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

The **one** and **zero** laws:

$A \cup \emptyset = A$ $\qquad A \cap \emptyset = \emptyset$.

The symmetrical appearance of these laws is no coincidence (it also makes them easier to remember). You might like to compare these laws with those of arithmetic, if $\cup$ is replaced by $+$, $\cap$ by $\times$, and $\emptyset$ by 0 (which of the laws do *not* hold under this analogy?).

**Proof** The laws are established by proving that they are correct. Looking

at the definitions of union and intersection, we see that the idempotence and commutativity laws are immediate and do not require a proof. We will include here only proofs of one each of the associativity and distributivity laws.

Let us prove $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$. The implicit hypothesis is simply that $A$, $B$ and $C$ are sets, so assume this. Recall that, to prove equality of sets, we usually need a double inclusion proof.

First, suppose that $x \in A \cup (B \cap C)$. This means that either $x \in A$ or $x \in B \cap C$. Now we must do a *proof by cases*: we know that one of two things is true, and we must show that either case leads to the desired conclusion. The two cases are

(i) $x \in A$. Then $x \in A \cup B$ (by the definition of $\cup$) and $x \in A \cup C$ (ditto), so $x \in (A \cup B) \cap (A \cup C)$.

(ii) $x \in B \cap C$. By the definition of $\cap$, $x \in B$ so $x \in A \cup B$. But also $x \in C$ so $x \in A \cup C$. Putting these together, $x \in (A \cup B) \cap (A \cup C)$.

So either way, $x \in (A \cup B) \cap (A \cup C)$. We have proved that $A \cup (B \cap C) \subseteq (A \cup B) \cap (A \cup C)$.

Second, suppose that $x \in (A \cup B) \cap (A \cup C)$. Then we know $x \in A \cup B$ and $x \in A \cup C$. Now unfortunately there are two sets of two cases, making a total of four possibilities:

(i) $x \in A$ and $x \in A$. Then $x \in A \cup (B \cap C)$.

(ii) $x \in A$ and $x \in C$. Using just the former, $x \in A \cup (B \cap C)$.

(iii) $x \in B$ and $x \in A$. Using just the latter, $x \in A \cup (B \cap C)$.

(iv) $x \in B$ and $x \in C$. Together they tell us that $x \in B \cap C$, so $x \in A \cup (B \cap C)$.

So in any case $x \in A \cup (B \cap C)$. We have proved that $A \cup (B \cap C) \supseteq (A \cup B) \cap (A \cup C)$.

Between these two halves, we have shown that $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$, completing the proof of one of the distributive laws.

Now let us try to prove $A \cap (B \cap C) = (A \cap B) \cap C$. This time we can avoid the longwinded double inclusion proof, but writing a proof which works both

ways:

$$\begin{aligned} x \in A \cap (B \cap C) &\Leftrightarrow x \in A \text{ and } x \in B \cap C \\ &\Leftrightarrow x \in A \text{ and } x \in B \text{ and } x \in C \\ &\Leftrightarrow x \in A \cap B \text{ and } x \in C \\ &\Leftrightarrow x \in (A \cap B) \cap C \end{aligned}$$

The symbol "$\Leftrightarrow$" means that one side is true **if and only if** the other is. More practically, it means that each step follows from the previous step, but also each step follows from the next step too. ■

Once proved, the algebraic laws allow the possibility of shorter proofs, avoiding having to re-prove everything from scratch. Sometimes a proof can be carried out entirely using algebraic laws, moving from the LHS (left hand side) of an equation and using laws to reach the RHS (right hand side) or vice versa. For example,

$$\begin{aligned} &(A \cup B) \cap (C \cup D) \\ &\quad = ((A \cup B) \cap C) \cup ((A \cup B) \cap D) && \{\text{distributivity}\} \\ &\quad = (C \cap (A \cup B)) \cup (D \cap (A \cup B)) && \{\text{commutativity}\} \\ &\quad = ((C \cap A) \cup (C \cap B)) \cup ((D \cap A) \cup (D \cap B)) && \{\text{distributivity}\} \\ &\quad = (C \cap A) \cup (C \cap B) \cup (D \cap A) \cup (D \cap B) && \{\text{associativity}\} \\ &\quad = (A \cap C) \cup (B \cap C) \cup (A \cap D) \cup (B \cap D) && \{\text{commutativity}\} \end{aligned}$$

while not exactly simple, is shorter than a formal proof from scratch of the same statement.

The **associative** law is particularly useful. It tells us that we need not include parentheses in $A \cup B \cup C$, because whether they are inserted as $(A \cup B) \cup C$ or $A \cup (B \cup C)$ it gives the same set. Using the associative law two or three times, the same is true for $A \cup B \cup C \cup D$ (we used this implicitly in the preceding proof), and the same law also holds for $\cap$. This allows us to use a shorthand for combining the elements of more than two sets:

**Definition** The **union** of the collection of sets $A_1, A_2, \ldots, A_n$ is the set whose elements are in any of the $A_i$:

$$\bigcup_{i=1}^{n} A_i = \{x \mid x \in A_i \text{ for some } i\}$$

The **intersection** of the collection of sets $A_1, A_2, \ldots, A_n$ is the set whose elements are in all of the $A_i$:

$$\bigcap_{i=1}^{n} A_i = \{x \mid x \in A_i \text{ for every } i\}$$

The same notation can apply to infinite sequences of sets: elements in any of $A_1, A_2, \ldots$ make up $\bigcup_{i=1}^{\infty} A_i$ and similarly for intersection. Because of the **commutative** and **idempotence** laws, it does not matter what order the $A_i$ come in or whether there are repetitions. This allows us to generalise further, to the intersection or union of a set of sets $\{A_i \mid i \in I\}$ for some **indexing set** $I$, written $\bigcup_{i \in I} A_i$.

For example, if we set $M_i = \{i, 2i, 3i, \ldots\}$ (all positive multiples of $i$) then

$$\bigcap_{i=1}^{4} M_i = \{12, 24, 36, 48, \ldots\} = M_{12}, \qquad \bigcup_{i=1}^{4} M_i = \mathbb{N}_+$$

$$\bigcup_{i=2}^{4} M_i = \{2, 3, 4, 6, 8, 9, 10, 12, 14, 15, \ldots\},$$

but

$$\bigcap_{i \in \mathbb{N}_+} M_i = \emptyset.$$

## 1.4 Complements, Symmetric Difference, Laws

There are other operations on sets, in addition to union and intersection. They relate to differences between sets:

**Definition** The **relative complement** of $B$ in $A$, written $A \setminus B$, is the set whose elements are in $A$ but not $B$:

$$A \setminus B = \{x \mid x \in A \text{ and } x \notin B\}.$$

The **symmetric difference** of $A$ and $B$, written $A \oplus B$, is the set whose elements are in one, but not both, of $A$ and $B$:

$$A \oplus B = \{x \mid (x \in A \text{ and } x \notin B) \text{ or } (x \in B \text{ and } x \notin A)\}.$$

In some books, the symmetric difference is written $A \,\Delta\, B$.

As a simple example, take $A = \{1, 3, 4\}$ and $B = \{3, 5, 7\}$. Then

$$A \cup B = \{1, 3, 4, 5, 7\} \qquad A \cap B = \{3\}$$
$$A \setminus B = \{1, 4\} \qquad A \oplus B = \{1, 4, 5, 7\}$$

There are very many algebraic laws relating $\cup$, $\cap$, $\setminus$, and $\oplus$: too many to list them all. Here are some of the most important:

**Claim 1.2** For any sets $A$, $B$ and $C$,
The **cancellation** laws:
$A \setminus A = \emptyset$, $\qquad A \setminus \emptyset = A$.
The **involution** law:
$A \setminus (A \setminus B) = A \cap B$.
**De Morgan's** laws:
$A \setminus (B \cup C) = (A \setminus B) \cap (A \setminus C)$, $\qquad A \setminus (B \cap C) = (A \setminus B) \cup (A \setminus C)$.
The **right-distributive** laws:
$(A \cup B) \setminus C = (A \setminus C) \cup (B \setminus C)$, $\qquad (A \cap B) \setminus C = (A \setminus C) \cap (B \setminus C)$.

(You will investigate some algebraic laws for $\oplus$ in the tutorial exercises.)

Finally, there is a useful special case of the relative complement construction. If all the sets we are interested in (for a particular problem, say) are subsets of a set $\mathcal{U}$ then we call $\mathcal{U}$ a **universe**. For example, if we are considering only positive numbers then $\mathcal{U} = \mathbb{N}_+$ could be a choice of universe.

**Definition** If $A \subseteq \mathcal{U}$ then we write $\overline{A}$ for $\mathcal{U} \setminus A$. This is called the **complement** of $A$ and is $\{x \mid x \notin A\}$ (under the assumption that all things under consideration are members of the universe).

(Some books use $A'$ or $A^c$ for the complement of $A$.)

The algebraic laws for complements can be derived from those for relative complements in Claim 1.2. They include the involution law $\overline{\overline{A}} = A$ and De Morgan's laws

$$\overline{A \cup B} = \overline{A} \cap \overline{B}, \quad \overline{A \cap B} = \overline{A} \cup \overline{B}.$$

## 1.5 Products and Power Sets

The set operations we have seen only form sets whose members are those of previously defined sets. The members themselves are not altered, simply included or excluded from the result. Two other operations – product and power set – create sets with other members.

**Definition** We use the symbol $(x, y)$ to mean the **ordered pair** of $x$ and $y$. $(x, y) = (x', y')$ if and only if $x = x'$ and $y = y'$. Although this is not a set, we still use the word **element** in the following context: $x$ is the first element of $(x, y)$, and $y$ is the second element. The word **component** is also used in this way.

If $A$ and $B$ are sets then the **cartesian product** of $A$ and $B$, written $A \times B$, is the set of pairs whose first element is from $A$ and second is from $B$, in any combination:

$$A \times B = \{(x, y) \mid x \in A \text{ and } y \in B\}.$$

An example of a simple cartesian product is $\{1, 3, 5\} \times \{2, 4, 6\} = \{(1, 2), (1, 4), (1, 6), (3, 2), (3, 4), (3, 6), (5, 2), (5, 4), (5, 6)\}$.

Cartesian products also obey some algebraic laws. The most important are the distributive laws:

**Claim 1.3**

$$A \times (B \cup C) = (A \times B) \cup (A \times C)$$
$$A \times (B \cap C) = (A \times B) \cap (A \times C)$$

**Proof** We just prove the first one. A double inclusion proof is necessary.

First, suppose that $(x, y) \in A \times (B \cup C)$. We know that only pairs can be elements of a cartesian product, which is why we can write it as $(x, y)$ right from the start. By the definition of cartesian product, we must have $x \in A$ and $y \in B \cup C$. Either

(i) $y \in B$, in which case $(x, y) \in A \times B$ and therefore $(x, y) \in (A \times B) \cup (A \times C)$, or

(ii) $y \in C$, in which case $(x, y) \in A \times C$ and therefore $(x, y) \in (A \times B) \cup (A \times C)$.

We have shown that $A \times (B \cup C) \subseteq (A \times B) \cup (A \times C)$.

Second, suppose that $(x, y) \in (A \times B) \cup (A \times C)$; again, only pairs can be members of this set. Then either

(i) $(x, y) \in A \times B$, in which case $x \in A$ and $y \in B$ and therefore $y \in B \cup C$, so $(x, y) \in A \times (B \cup C)$, or

(ii) $(x, y) \in A \times C$, in which case $x \in A$ and $y \in C$ and therefore $y \in B \cup C$, so $(x, y) \in A \times (B \cup C)$.

We have shown that $A \times (B \cup C) \supseteq (A \times B) \cup (A \times C)$, completing the double inclusion proof. ■

(In practice we would not normally write out all the details of proofs like this. When two cases are "symmetrical" – the same, with some letter or symbol swapped for another, as in both of the proof-by-cases above – then it is sensible to include only one and say that the other follows "by symmetry".)

We can extend ordered pairs to ordered triples and more generally to ordered **n-tuples** $(x_1, x_2, \ldots, x_n)$. The cartesian product extends to finite products $\bigtimes_{i=1}^{n} A_i$. There are two minor differences between the multiple form of the cartesian product and those of union and intersection we saw earlier. First, we generally only allow finite products. Second, the standard cartesian product is *not associative* because $A \times (B \times C)$ contains elements of the form $(a, (b, c))$ (where $a \in A$ etc.) but $(A \times B) \times C$ contains elements of the form $((a, b), c)$. These are not equal elements, because their first and second components are not the same, nor is either equal to the 3-tuple $(a, b, c)$. However there is a natural correspondence between all these elements and so it is quite common to pretend that they are all the same.

Furthermore, the cartesian product is *not commutative* because the elements of $A \times B$ are reversed compared with $B \times A$. Even though there is a natural correspondence between them, it is usual to keep these sets distinct.

It is convenient to write $A^2$ for $A \times A$, $A^3$ for $A \times A \times A$, and so on.

Cartesian products occur particularly often in computer science specifications, combining multiple observations into one compound observation. Suppose that a program involves two counters, which can only be positive integers, and an accumulator takes positive or negative integers. We might represent the **state** of the program at any instant by an element of $\mathbb{N}_+ \times \mathbb{N}_+ \times \mathbb{Z}$, with the first two elements representing the values of the counters and the last element representing the accumulator.

Finally, we have the power set:

**Definition** If $A$ is a set then the **power set** of $A$, written $\mathcal{P}(A)$, is the set of all subsets of $A$:

$$\mathcal{P}(A) = \{B \mid B \subseteq A\}.$$

(Remember that this includes the case $B = A$).

Power sets rapidly become large and complex. For example, $\mathcal{P}(\emptyset) = \{\emptyset\}$, $\mathcal{P}(\mathcal{P}(\emptyset)) = \{\emptyset, \{\emptyset\}\}$, $\mathcal{P}(\mathcal{P}(\mathcal{P}(\emptyset))) = \{\emptyset, \{\emptyset\}, \{\{\emptyset\}\}, \{\emptyset, \{\emptyset\}\}\}$, .... In general, $\mathcal{P}(\{1, 2, \ldots, n\})$ consists of $2^n$ elements, in which each of $1, 2, \ldots, n$ is included or not included, in every combination.

## 1.6 Cardinality

The *size* of a set is called its **cardinality**, and it is common to write $\#A$ or $|A|$ for the cardinality of $A$. When $A$ is a finite set this is simply the number of elements: $|\emptyset| = 0$ and $|\{1, 2, 4, 8, 16\}| = 5$. Quite a substantial part of discrete mathematics is involved with counting the cardinality of finite sets, and we will learn techniques for doing so in Chapter 3.

In the tutorial exercises you will be asked how the cardinality of combined sets $A \cup B$, $A \cap B$, $A \setminus B$, $A \oplus B$, $A \times B$, and $\mathcal{P}(A)$ depend on the cardinality of $A$ and $B$.

In this course we shall only consider the cardinality of finite sets. You should know that cardinality can be extended to infinite sets, and not all infinite sets have the same cardinality (some infinite sets are "bigger" than others).

## 1.7 Interesting Diversion: Bags

In computer science, it is often useful to consider unordered collections of objects where duplicates are allowed. These are known as **bags**. Some authors write bags in the same way as sets, but some use special bag-brackets. For example $\wr 1, 2, 3, 3 \wr$ defines a bag in which the elements 1 and 2 occur once, and 3 occurs twice. It is identical to $\wr 3, 1, 2, 3 \wr$ but not $\wr 1, 2, 3 \wr$.

Bags are not uniquely defined by knowing which elements are and are not members: the number of occurrences also matters. The simplest way to

formalize bags is to use a function to count the number of occurrences of each element (functions are the subject of the next chapter).

Most of the same operations can be defined for bags as for sets. The **union** of two bags adds the number of occurrences of each element; the **intersection** takes the minimum number of occurrences of each. For example, $\wr 1, 2, 3, 3\wr \cup \wr 2, 3, 4\wr = \wr 1, 2, 2, 3, 3, 3, 4\wr$ while $\wr 1, 2, 3, 3\wr \cap \wr 2, 3, 4\wr = \wr 2, 3\wr$. Bag difference subtracts elements (although, of course, there can never be a negative number of occurrences of any element): $\wr 1, 2, 3, 3\wr \backslash \wr 2, 3, 4\wr = \wr 1, 3\wr$.

Some of the algebraic laws for sets also hold for bags: commutatively and associativity of union and intersection, for example. Others fail: idempotence ($\wr 1\wr \cup \wr 1\wr = \wr 1, 1\wr$, not $\wr 1\wr$), and (importantly) some of the distributivity laws ($\wr 1\wr \cap (\wr 1\wr \cup \wr 1\wr) = \wr 1\wr$, not $\wr 1, 1\wr$) are false.

Few mathematics books mention bags, but they are useful to computer scientists and are particularly relevant to the theory of databases.

---

## Practice Questions

**1.1** Which of the following statements are true?

| | | | | | |
|---|---|---|---|---|---|
| (i) | $\emptyset \in \emptyset$, | (ii) | $\emptyset \subset \emptyset$, | (iii) | $\emptyset \subseteq \emptyset$, |
| (iv) | $\emptyset \in \{\emptyset\}$, | (v) | $\emptyset \subset \{\emptyset\}$, | (vi) | $\emptyset \subseteq \{\emptyset\}$, |
| (vii) | $\emptyset \in \{\{\emptyset\}\}$, | (viii) | $\emptyset \subset \{\{\emptyset\}\}$, | (ix) | $\emptyset \subseteq \{\{\emptyset\}\}$. |

**1.2** Let $A = \{3, 2, 1\}$ and $B = \{3, 4\}$ and suppose that the universe $\mathcal{U} = \{1, 2, 3, 4, 5\}$. Compute

| | | | | | |
|---|---|---|---|---|---|
| (i) | $A \cup B$, | (ii) | $A \cap B$, | (iii) | $A \setminus B$, |
| (iv) | $B \setminus A$, | (v) | $A \oplus B$, | (vi) | $A \times B$, |
| (vii) | $\mathcal{P}(A)$. | (viii) | $\overline{A}$, | (ix) | $\overline{B}$, |
| (x) | $\overline{A \cup B}$, | (xi) | $\overline{A} \cap \overline{B}$. | | |

Write down the cardinality of each of the above sets.

**1.3** For each $i \in \mathbb{N}_+$, let $A_i = \{-i, -i+1, \ldots, -1, 0, 1, \ldots, i-1, i\}$. What are $\bigcup_{i=1}^{\infty} A_i$ and $\bigcap_{i=1}^{\infty} A_i$?

**1.4** To prove that something is not true it is usually necessary to give a **counterexample**: a concrete example which demonstrates the falsity of a statement. The following statements are not, in general, true; find counterexamples.

(i) $A \cap \emptyset = A$.
(ii) $A \setminus B = B \setminus A$.
(iii) $A \setminus (B \cup C) = (A \setminus B) \cup (A \setminus C)$.
(iv) $A \cap (B \cup C) = (A \cap B) \cup C$.

(Quite often, false statements have very simple counterexamples so it makes sense to look at small sets.)

**1.5** Using only the algebraic laws in Claims 1.1 and 1.2, prove that

$$(A \setminus B) \cup (B \setminus A) = (A \cup B) \setminus (A \cap B)$$

(This gives two equivalent formulae for $A \oplus B$.)

**1.6** Prove from scratch that $A \subseteq B$ and $B \subseteq C$ together imply $A \subseteq C$.

**1.7** Prove that whenever $A \subseteq B$, $\mathcal{P}(A) \subseteq \mathcal{P}(B)$.

**1.8** The equation $\mathcal{P}(A \cup B) = \mathcal{P}(A) \cup \mathcal{P}(B)$ is not always true. Find a counterexample.

## Answers to Chapter 1 Practice Questions

**1.1** (iii), (iv), (v), (vi), (viii) and (ix) are true.

**1.2** (i) $\{1,2,3,4\}$, (ii) $\{3\}$, (iii) $\{1,2\}$, (iv) $\{4\}$, (v) $\{1,2,4\}$,
(vi) $\{(1,3),(1,4),(2,3),(2,4),(3,3),(3,4)\}$,
(vii) $\{\emptyset,\{1\},\{2\},\{3\},\{1,2\},\{2,3\},\{1,3\},\{1,2,3\}\}$,
(viii) $\{4,5\}$, (ix) $\{1,2,5\}$, (x) $\{5\}$, (xi) $\{5\}$.
Their cardinalities are (in order): 4,1,2,1,3,6,8,2,3,1,1.

**1.3** $\mathbb{Z}$ and $\{-1,0,1\}$.

**1.4** There are many answers; the simplest, but perhaps not most informative, are (i) $A=\{1\}$; (ii) $A=\emptyset$, $B=\{1\}$; (iii) $A=\{1\}$, $B=\{1\}$, $C=\emptyset$; (iv) $A=\emptyset$, $B=\emptyset$, $C=\{1\}$. In each case, the LHS of the statement equals $\emptyset$ and the RHS equals $\{1\}$.

**1.5** Starting from the RHS, we have

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

which equals the LHS. At each stage we used (sometimes more than one law per step): De Morgan's law; right-distributivity of $\setminus$ over $\cup$; associativity of $\cup$ and cancellation of $\setminus$; zero and commutativity of $\cup$.

**1.6** Suppose that $A\subseteq B$ and $B\subseteq C$; we must now show that $x\in A$ implies $x\in C$. Whenever $x\in A$, we must have $x\in B$ (because of the first assumption) and therefore $x\in C$ (because of the second). This completes the proof.

**1.7** We must show that whenever $C\in\mathcal{P}(A)$, $C\in\mathcal{P}(B)$. So suppose that $C\in\mathcal{P}(A)$; this means that $C\subseteq A$, and therefore $C\subseteq B$ (by the previous exercise), which is exactly what we needed.

**1.8** In searching for a counterexample, you can rule out looking at cases when the equation is known to be true. By the previous two exercises and the algebraic laws for $\cup$, the given statement is true at least whenever $A\subseteq B$ or vice versa. So try $A=\{1\}$ and $B=\{2\}$. $\mathcal{P}(A\cup B)=\{\emptyset,\{1\},\{2\},\{1,2\}\}$ but $\mathcal{P}(A)\cup\mathcal{P}(B)=\{\emptyset,\{1\},\{2\}\}$; these two sets do not have all the same elements so they are not equal.

# Chapter 2

# Functions

---

**Reading:** Ross & Wright: 1.5, 1.7;
Chetwynd & Diggle: 1.5, 3.5–3.8;
Grimaldi: 5.2, 5.3, 5.6;
Grossman: 6.1, 6.2.

---

Functions, and their more general counterparts relations (which we will meet in Chapter 4), are ways of associating elements of one set with elements of another. You are probably used to functions as "operations" on sets and we will introduce them in the same style. We will also meet two (related) proof techniques: proof of the contrapositive, and proof by contradiction.

## 2.1 Intervals

Before we introduce functions, though, it will be convenient to define some new sets. They are subsets of $\mathbb{R}$ with the **interval property**:

**Definition** For $I \subseteq \mathbb{R}$, $I$ is an **interval** if, whenever $x, z \in I$ with $x < y < z$ then $y \in I$.

This means that there are no "gaps" in the interval. It is important to note that intervals are defined to be subsets of $\mathbb{R}$ (not $\mathbb{Z}$ or $\mathbb{Q}$).

There are some useful shorthands associated with intervals.

**Definition** If $a, b \in \mathbb{R}$ and $a < b$ we define the following sets:

$$\begin{aligned} (a,b) &= \{x \in \mathbb{R} \mid a < x < b\} & [a,b] &= \{x \in \mathbb{R} \mid a \leq x \leq b\} \\ (a,b] &= \{x \in \mathbb{R} \mid a < x \leq b\} & [a,b) &= \{x \in \mathbb{R} \mid a \leq x < b\} \\ (a,\infty) &= \{x \in \mathbb{R} \mid a < x\} & [a,\infty) &= \{x \in \mathbb{R} \mid a \leq x\} \\ (-\infty,b) &= \{x \in \mathbb{R} \mid x < b\} & (-\infty,b] &= \{x \in \mathbb{R} \mid x \leq b\} \end{aligned}$$

The idea is that "round brackets" indicate that the endpoint is not included, and "square brackets" that the endpoint is included. Remember that $\infty$ and $-\infty$ are not elements of $\mathbb{R}$ – they are not numbers at all – so they could never be included in an interval. $(-\infty, \infty)$ is another notation for the whole set $\mathbb{R}$, and $[a, a]$ is the singleton set $\{a\}$. The notation $(a, a)$ is not used, but if it were then it would represent the empty set.

Intervals $(a, b)$, $(a, \infty)$ or $(-\infty, b)$ are called **open intervals**. Those of the form $[a, b]$ are called **closed intervals** and those with mixed brackets are called **half-open intervals**.

## 2.2 Definition of Functions

The traditional definition of function has a rather abstract formulation. We will postpone that until you have seen **relations** in Chapter 4, and adopt a simpler definition for now.

**Definition** A **function** consists of three parts:

(i) A set $A$ called the **domain**,

(ii) A set $B$ called the **codomain**,

(iii) A map which associates every element of $A$ with exactly one element of $B$.

We can define a function $f$ by specifying the three components, and we usually do this in the following notation. $f : A \to B$ deals with the first two, specifying that $f$ is to be a function whose domain is $A$ and codomain is $B$ (when $A = B$ we can say that $f$ is a **function on** $A$). Then we write either $f : a \mapsto b$ or $f(a) = b$, to say that $f$ associates the element $a \in A$ with $b \in B$. Either we must give every such map, one for each element of $A$, or write a formula for $b$ in terms of $a$.

Given a function $f$, we write $\mathrm{Dom}(f)$ for the domain of $f$.

Simple examples of functions include $f : \mathbb{R} \to [0, \infty)$, $f : x \mapsto x^2$ (which defines the squaring function on the real numbers), $g : \mathbb{N} \to \mathbb{N}$, $g : n \mapsto n+1$ (the add-one function on the natural numbers), $h : \mathbb{N}_+ \to \mathbb{N}_+$, $h : n \mapsto 1 \cdot 2 \cdot 3 \cdots n$ (the factorial function, usually extended to $\mathbb{N}$ by setting $h(0) = 1$), for *any* set $A$ the function $\mathrm{id}_A : A \to A$ given by $\mathrm{id}_A : x \mapsto x$ (the so-called **identity function** on $A$), and more complex mathematical functions such as $\sin : \mathbb{R} \to \mathbb{R}$.

Two functions $f$ and $g$ are **equal** if all three of their components are the same: the domain of $f$ must match the domain of $g$, likewise for the codomain, and we need $f(a) = g(a)$ for every $a \in \mathrm{Dom}(f)$.

We can think of the elements of the domains as *inputs* to $f$, and those of the codomain as the *outputs*. In practice, we often use functions with more than one input. Formally, these are functions whose domain is a cartesian product. For example the function which performs addition of real numbers is $+ : \mathbb{R}^2 \to \mathbb{R}$. There is a little more on functions with two inputs in Section 2.6.

It is important that a function must associate every element of $A$ with exactly one element of $B$. It is not allowed for a function to be "multi-valued" (although some mathematics books on complex analysis do consider so-called multifunctions), neither is it allowed for a function to be undefined at some points of $A$. So $f : \mathbb{R} \to \mathbb{R}$, $f : x \mapsto \log x$ is not a function because it is not defined for negative numbers. Nonetheless, in computer science it can be useful to discuss functions which are not defined everywhere on their domain, so we introduce

**Definition** A **partial function** has a domain $A$ and codomain $B$, and associates every element of $A$ with at most one element of $B$.

When it is important to stress that a function is *not* partial, i.e. it is defined everywhere on the domain, we say that it is a **total function**.

An example often found in computing is the partial function which associates the input of some particular program with its output (to do so we must express the possible inputs and outputs as sets, but they do not have to be sets of numbers). Because some programs fail to produce an output, this function may well be partial.

## 2.3 Properties of Functions

It is not necessary for every element of $B$ to be involved in the map $f : A \to B$. We can pick out those which are possible "outputs" of $f$:

**Definition** If $f : A \to B$ then the **image** of $f$, written $\mathrm{Im}(f)$, is the set $\{b \in B \mid f(a) = b \text{ for some } a \in A\}$.

(Another way of writing this is $\{f(a) \mid a \in A\}$.)

The image of $f$ is also sometimes referred to as the **image of** $A$ **under** $f$. Some books use the word **range** instead of image.

Using the examples of the last section, $f : \mathbb{R} \to [0, \infty)$, $f : x \mapsto x^2$ has $\mathrm{Im}(f) = [0, \infty)$; $g : \mathbb{N} \to \mathbb{N}$, $g : n \mapsto n + 1$ has $\mathrm{Im}(g) = \mathbb{N}_+$; $h : \mathbb{N}_+ \to \mathbb{N}_+$, $h : n \mapsto 1 \cdot 2 \cdot 3 \cdots n$ has $\mathrm{Im}(h) = \{1, 2, 6, 24, 120, \ldots\}$. The image of $\sin : \mathbb{R} \to \mathbb{R}$ is the interval $[1, 1] = \{x \in \mathbb{R} \mid -1 \leq x \leq 1\}$.

Now some properties which reflect how accurately the function $f : A \to B$ matches up the sets $A$ and $B$:

**Definition** Let $f : A \to B$ be a function.

We say that $f$ is **onto** if every element of $B$ is a possible output of $f$; formally, if for every $b \in B$ there is some $a \in A$ with $f(a) = b$. (This is equivalent to saying: $\mathrm{Im}(f) = B$.)

We say that $f$ is **1-1** if no element of $B$ is the output corresponding to two or more distinct inputs; formally, if for every distinct pair $a_1, a_2 \in A$ we have $f(a_1) \neq f(a_2)$.

We say that $f$ is **bijective** if it is both onto and 1-1.

In some books the terminology is different:

- **surjective** is used instead of onto, and an onto function is a **surjection**.
- **injective** is used instead of 1-1, and a 1-1 function is an **injection**.
- A bijective function is called an **bijection**.

Although strictly speaking a bijection is a function *from* $A$ *to* $B$, because of the results of Section 2.5 it is common to say that a bijection is *between* $A$ *and* $B$. The phrase **one-to-one map** is also sometimes used to mean a bijection, but this is too easily confused with "1-1".

Here are some examples. On any set $A$, the identity function $f(x) = x$ is both onto and 1-1 (so it is a bijection). The function $g : \mathbb{N} \to \mathbb{N}$, $g : n \mapsto n+1$ is 1-1 because $n_1 + 1 \neq n_2 + 1$ if $n_1 \neq n_2$, but it is not onto because there is no $n \in \mathbb{N}$ with $n + 1 = 0$. On the other hand, the very similar function $g' : \mathbb{Z} \to \mathbb{Z}$, $g' : n \mapsto n + 1$ *is* onto as well as 1-1, so it is bijective.

We see that the domain and codomain are very important to knowing whether a function is 1-1, or onto, or both. The function $h : \mathbb{R} \to \mathbb{R}$, $h : x \mapsto x^2$ is neither 1-1 (because $x$ and $-x$ square to the same number) nor onto (because no real number squares to a negative number) but the similar function $h' : [0, \infty) \to [0, \infty)$, $h' : x \mapsto x^2$ is both 1-1 and onto. By reducing the codomain to match the image, a function can always be made onto; it is also possible, but usually not desirable, to cut down the domain of a function to make it 1-1.

If $f : A \to B$ is a bijection, this tells us interesting things about the sets $A$ and $B$: their elements are paired up, so $|A| = |B|$ (indeed, this is part of the definition of cardinality for infinite sets). This fact can be helpful in counting the number of members of a set $A$: sometimes it is easier to count the members of a different set $B$ and then match up the elements by finding a bijection between $A$ and $B$. Techniques for counting sets will be considered in Chapter 3.

## 2.4 The Contrapositive and Proof by Contradiction

Putting aside functions for a moment, we need to introduce a little bit of logic. Most interesting statements are of the form "if A then B", which we also write $A \Rightarrow B$. The **contrapositive** statement is "if not B then not A", or $\neg B \Rightarrow \neg A$. It is important to know that these two statements are equivalent.

It is a very common mistake to confuse $A \Rightarrow B$ and $B \Rightarrow A$. They are *not* the same. A simple example is the statement "if it is raining, then there are clouds overhead" (let us say that this is true). This is not the same as "if there are clouds overhead, then it is raining" (which we can agree is not true). On the other hand, the contrapositive of the first statement is "if there are no clouds overhead, it cannot be raining" (true).

Because $A \Rightarrow B$ and the contrapositive $\neg B \Rightarrow \neg A$ are equivalent, proving one is sufficient to prove the other. Sometimes it is easier to prove the contrapositive of the statement you want. This is particularly applicable to proving that a function is 1-1: the definition says that $a_1 \neq a_2 \Rightarrow f(a_1) \neq f(a_2)$, but it is often convenient to prove the contrapositive $f(a_1) = f(a_2) \Rightarrow a_1 = a_2$.

Related to the idea of contrapositive is a proof technique called **proof by contradiction**. Suppose that, under some hypotheses, we want to prove a statement $S$. We *assume* that $S$ is *false*, and then continue proving things, trying to find a contradiction (something which is clearly false). Once we have got to the contradiction we deduce that the assumption that $S$ was false must have been wrong, and so $S$ is true.

The classic example of a proof by contradiction is of $\sqrt{2} \notin \mathbb{Q}$. Recall that $\mathbb{Q}$ is the set of **rational numbers** (those which can be represented as a fraction); numbers in $\mathbb{R} \setminus \mathbb{Q}$ are called **irrational**.

**Claim 2.1** $\sqrt{2}$ is irrational.

**Proof** Let us suppose, for a contradiction, that $\sqrt{2} \in \mathbb{Q}$. Then

$$\sqrt{2} = m/n \tag{2.1}$$

for some integers $m$ and $n$, and furthermore we can suppose that the fraction is in "lowest terms", so $m$ and $n$ have no common factors. Multiplying the equation (2.1) by $n$ and squaring gives

$$2n^2 = m^2 \tag{2.2}$$

so $m^2$ must be even, which means that $m$ must be even (proof of the contrapositive: if $m$ is odd then $m^2$ is the product of odd numbers and therefore odd). Let us write $m = 2m'$, and substitute into (2.2), giving

$$n^2 = 2m'^2$$

so $n$ is also even, and this is now a contradiction because we shown that 2 is a common factor of $m$ and $n$, yet $m$ and $n$ had no common factors. So the assumption that $\sqrt{2} \in \mathbb{Q}$ must be wrong: we have proved that $\sqrt{2}$ is irrational. ■

Now for a related proof by contradiction, which is relevant to the topic of this chapter:

**Claim 2.2** The function $f : \mathbb{Q} \to \mathbb{Q}$, $f(x) = x^5 + x^3$ is not onto.

**Proof** It is worth noting that $f$ is **well-defined** because the output is always rational when the input is rational (sums, products, and integer powers of rational numbers are always rational).

To prove that $f$ is not onto it is only necessary to find one example of a rational number $y$ such that $f(x) \neq y$ for any $x$. $f$ can take arbitrarily large or small values, so we must look for a "gap" in the image (which is difficult because there is no closed form for determining $x$ from $f(x)$). $y = 0$ will not do, because $f(0) = 0$; $y = 2$ will not do either, because $f(1) = 2$. We will prove that $y = 1$ has the required property.

Suppose, then, for a contradiction that there is some $x \in \mathbb{Q}$ with $f(x) = 1$. In other words,

$$\left(\tfrac{m}{n}\right)^5 + \left(\tfrac{m}{n}\right)^3 - 1 = 0 \tag{2.3}$$

for some $m, n \in \mathbb{Z}$ and we may assume, as before, that $m$ and $n$ have no common factors. Multiplying through by $n^5$ gives

$$m^5 + m^3n^2 - n^5 = 0. \tag{2.4}$$

Now what is the **parity** of $m$ and $n$? (The parity of a number is whether it is even or odd). There are four cases:

(i) $m$ and $n$ are both odd. But then $m^5$ is odd, $m^3n^2$ is odd, and $n^5$ is odd, so the left of (2.4) is odd. The right side is even, so this is a contradiction.

(ii) $m$ is odd and $n$ is even. But then $m^5$ is odd, $m^3n^2$ is even, and $n^5$ is even, and the same argument applies.

(iii) $m$ is even and $n$ is odd. But then $m^5$ is even, $m^3n^2$ is even, and $n^5$ is odd, and the same argument applies.

(iv) $m$ is even and $n$ is even. But $m$ and $n$ are supposed to have no common factors, so this is a contradiction.

In each case there is a contradiction. So the assumption that $x \in \mathbb{Q}$ must be false. This completes the proof that $f$ is not onto, because 1 is not in the image of $f$. ■

Proofs by contradiction can seem a bit strange, because we write down statements which are false (normally we only write true statements in a proof).

One finds oneself writing down more and more bizarre-looking mathematics, until the contradiction is so obvious that we can finish the proof.

The similarity between proof of the contrapositive and proof by contradiction is that they both begin by assuming the falsity of the (conclusion of the) statement we are trying to prove. Proof of the contrapositive tries to deduce that the hypotheses of the statement are false; proof by contradiction assumes that they are true, and tries to find a contradiction. As a result, it is often harder to construct a proof by contradiction, because one does not know "where to aim for". There is a school of thought which says that proofs by contradiction should be avoided unless there is no alternative.

## 2.5 Operations on Functions: Composition and Inverse

Applying one function to the result of another is called a composition:

**Definition** If $f : A \to B$ and $g : B \to C$ then their **composition** is the function $g \circ f : A \to C$ given by $(g \circ f)(x) = g(f(x))$.

$g \circ f$ can be pronounced "$g$ after $f$". In some fields of computer science it might be written $f; g$ ("$f$ then $g$"). It is important that the codomain of $f$ matches the domain of $g$ exactly, otherwise the functions cannot be composed.

For a simple example, consider $f : \mathbb{R} \to \mathbb{R}$ given by $f(x) = 2x + 3$ and $g : \mathbb{R} \to \mathbb{R}$ given by $g(x) = x^2$. Both compositions $g \circ f$ and $f \circ g$ exist, and both are functions on $\mathbb{R}$. We have $(g \circ f)(x) = (2x + 3)^2$ and $(f \circ g)(x) = 2x^2 + 3$.

The example illustrates that $g \circ f$ and $f \circ g$ need not be equal (to make this explicit we should give an element of $\mathbb{R}$ for which the values of the functions $g \circ f$ and $f \circ g$ disagree: $x = 0$ will do). So the composition operation is not necessarily **commutative**, although for some choices of $f$ and $g$ it is *possible* for $g \circ f$ and $f \circ g$ to be equal functions.

On the other hand,

**Claim 2.3** Composition of functions is **associative**. That is, if $f : A \to B$, $g : B \to C$, $h : C \to D$ then the compositions $h \circ (g \circ f)$ and $(h \circ g) \circ f$ both exist, and are equal functions.

**Proof** Because the domain of $g$ matches the codomain of $f$, and the domain of $h$ matches the codomain of $g$, all the compositions exist. Furthermore, $h \circ (g \circ f) : A \to D$ and $(h \circ g) \circ f : A \to D$ so it only remains to check that $h \circ (g \circ f)$ and $(h \circ g) \circ f$ agree on any input.

Let $a$ be any member of $A$. On one hand we have $(h \circ (g \circ f))(a) = h((g \circ f)(a)) = h(g(f(a)))$; on the other $((h \circ g) \circ f)(a) = (h \circ g)(f(a)) = h(g(f(a)))$, so the two functions are equal. ■

If we think of a function as an "action", "changing" elements of the domain into elements of the codomain, then it becomes natural to ask whether the effect of the action can be undone. That is, can we reverse the function? When this is possible (it is not always) then we have an inverse function:

**Definition** If $f : A \to B$ and $g : B \to A$ satisfy $g \circ f = \mathrm{id}_A$ and $f \circ g = \mathrm{id}_B$ then we say that $g$ is the **inverse function** of $f$ and write $g = f^{-1}$.

(We have written "the" inverse because it is a consequence of the definition, although we will not prove it, that there can only be at most one inverse for $f$.)

That is, $f^{-1}(f(a)) = a$ and $f(f^{-1}(b)) = b$ for each $a \in A$ and $b \in B$; it is necessary for both equations to hold. Simple examples include $f : \mathbb{Z} \to \mathbb{Z}$, $f : n \mapsto n + 1$ which has inverse $f^{-1} : \mathbb{Z} \to \mathbb{Z}$, $f : n \mapsto n - 1$; $g : [0, \infty) \to [0, \infty)$, and $g : x \mapsto x^2$, which has inverse $g^{-1} : x \mapsto \sqrt[+]{x}$.

To try to find an inverse for a function $f$ it is usual to write $y = f(x)$ and attempt to rearrange the equation to the form $x = g(y)$, in which case $g = f^{-1}$. For example, to find the inverse of $f : (0, 1) \to (1, \infty)$, $f(x) = 1/(1 - x)$ we write $y = 1/(1 - x)$, so $1/y = 1 - x$, so $x = 1 - 1/y$. This is well-defined on $(1, \infty)$ so $g : (1, \infty) \to (0, 1)$, $g(y) = 1 - 1/y$ is the inverse of $f$.

However, not all functions have an inverse. First, $f(f^{-1}(b)) = b$ is impossible if $b \notin \mathrm{Im}(f)$, so we know that $f$ cannot have an inverse if it is not onto. Second, if $f(a_1) = f(a_2)$ for $a_1 \neq a_2$ then necessarily $f^{-1}(f(a_1)) = f^{-1}(f(a_2))$ so we cannot have both $f^{-1}(f(a_1)) = a_1$ and $f^{-1}(f(a_2)) = a_2$. Therefore $f$ cannot have an inverse if it is not 1-1. Put together, we have proved that a function which has an inverse must be bijective. The converse is also true, although we will not prove it now. So

**A function $f : A \to B$ has an inverse if and only if $f$ is bijective.**

Finally, we introduce one other operation on functions, formalising the idea of cutting down a domain but keeping all the relevant parts of the map.

**Definition** If $f : A \to B$ is a function and $A' \subseteq A$ then the **restriction** of $f$ to $A'$ is $f{\restriction_{A'}}\colon A' \to B$ given by $f'(a) = f(a)$ for each $a \in A'$.

## 2.6 Interesting Diversion: Binary Operators

Functions of two inputs play a special role in both mathematics and computer science. A function $f : A \times A \to A$, where the possible values of both inputs matches that of the output, is called a **binary operator** on $A$. Examples include: addition and multiplication of numbers or matrices; union, intersection, relative complement, and symmetric difference of sets (as long as the sets are restricted to a particular universe); and concatenation of strings.

(Why must sets be restricted to a particular universe? It is a technicality. We may not define, for example, $\cup : S \times S \to S$ as the full union "function" because the domain and codomain of a function must be sets. $S$ would have to be the set of all sets and, it turns out, the collection of all sets is one of those objects which cannot itself be a set. But if given a universe $\mathcal{U}$ we can define union, intersection, and so on for subsets of $\mathcal{U}$ by taking $S = \mathcal{P}(\mathcal{U})$.)

It is quite common for binary operators to be written **infix**, meaning that the name of the function comes in between its two arguments, instead of in front of them (**prefix**). For example the addition operator on the real numbers $+ : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is usually written $x + y$ rather than $+(x, y)$. There are some particularly interesting properties which some binary operators possess:

**Definition** We say that a binary operator $\cdot$ on $A$:

- is **idempotent** if $x \cdot x = x$ for all $x \in A$.
- is **commutative** if $x \cdot y = y \cdot x$ for all $x, y \in A$.
- is **associative** if $(x \cdot y) \cdot z = x \cdot (y \cdot z)$ for all $x, y, z \in A$.
- **has an identity element** $e \in A$ if $e \cdot x = x \cdot e = x$ for all $x \in A$.

For subsets of some universal set, the union binary operator has all these properties (remember the algebraic laws for sets) with the identity element being $\emptyset$. The intersection binary operator also has all the properties, with the identity element being the universal set $\mathcal{U}$. Addition and multiplication of numbers is commutative, associative, and has an identity (zero in the case of addition, one in the case of multiplication) but not idempotent. Multiplication of matrices is associative and has an identity (the identity matrix) but not idempotent or commutative; the same is true of concatenation of strings.

Binary operators are equally important to the most abstract of mathematics as to computer science. Those which are associative and have an identity element are called **monoids**, and they occur often in the theory of programming.

---

## Practice Questions

**2.1** Which of the following are true?

(i) $\{0,1\} \subseteq (0,1)$, (ii) $\{0,1\} \subseteq [0,1]$,
(iii) $(0,1) \subseteq [0,1)$, (iv) $(0,1) \subseteq \mathbb{Q}$,
(v) $(0,1) = \bigcup_{i=1}^{\infty} \left(0, \frac{n-1}{n}\right)$, (vi) $(0,1) = \bigcap_{i=1}^{\infty} \left(-\frac{1}{n}, \frac{n+1}{n}\right)$.

**2.2** Show that

$$(A \setminus B) \cap (B \setminus A) = \emptyset$$

using a proof by contradiction.

**2.3** Why does $f : \mathbb{R} \to \mathbb{R}$, $f : x \mapsto 1/x$ not define a function?

**2.4** Which of the following functions are 1-1, and which are onto? For any which are bijective, compute the inverse function.

(i) $f : \mathbb{R} \to \mathbb{R}$, $f(x) = e^{-x}$.
(ii) $f : \mathbb{R} \to (0, \infty)$, $f(x) = e^{-x^2}$.
(iii) $f : \mathbb{R} \to [-1, 1]$, $f(x) = \cos x$.
(iv) $f : \mathbb{R} \to \mathbb{R}$, $f(x) = \begin{cases} 1/x, & \text{if } x \neq 0, \\ 0, & \text{if } x = 0. \end{cases}$

**2.5** For exactly which sets $A$ is the constant function $f : A \to \mathbb{N}$, $f(x) = 0$ injective? For which sets $A$ is it surjective?

**2.6** For which $n \in \mathbb{N}$ is the function $f : \mathbb{R} \to \mathbb{R}$, $f(x) = x^n$ injective? For which is it surjective?

**2.7** Let $f : A \to B$ and $g : B \to C$. Prove that if $f$ and $g$ are both onto, then $g \circ f$ is onto.

**2.8** Let $f : \mathbb{R} \to \mathbb{R}$, $f(x) = 3x + 1$, $g : \mathbb{R} \to \mathbb{R}$, $g(x) = (x-1)^3$. Compute

(i) $g \circ f$, (ii) $f^{-1}$, (iii) $g^{-1}$,
(iv) $(g \circ f)^{-1}$, (v) $f^{-1} \circ g^{-1}$.

## Answers to Chapter 2 Practice Questions

**2.1** (ii), (iii) and (v) are true.

**2.2** Suppose for a contradiction that $x \in (A \setminus B) \cap (B \setminus A)$. Therefore $x \in A \setminus B$ and $x \in B \setminus A$, so $x \in A$ and $x \notin B$ and $x \notin A$ and $x \notin B$. This is a contradiction, so the supposition that there was any $x$ in $(A \setminus B) \cap (B \setminus A)$ is false, so $(A \setminus B) \cap (B \setminus A) = \emptyset$.

**2.3** It does not define a value for $f(0)$.

**2.4** (i) 1-1 but not onto (does not produce negative values);
(ii) neither 1-1 nor onto ($x$ and $-x$ map to the same result, does not produce values greater than 1);
(iii) onto but not 1-1 ($x$ and $2k\pi \pm x$, for $k \in \mathbb{Z}$, map to the same result);
(iv) 1-1 and onto, and the function is its own inverse.

**2.5** $f(x) = 0$ is not injective if $f(a_1) = f(a_2)$ for distinct $a_1$ and $a_2$ in $A$; since this equation is always true, $f$ can only be injective if there are not a distinct pair of elements of $A$. Therefore $f$ is injective if and only if $A = \emptyset$ or $A = \{a\}$, a singleton set.
$f$ can never be surjective.

**2.6** It is both injective and surjective when $n$ is odd, and neither when $n$ is even.

**2.7** Assume that $f$ and $g$ are onto. (To show that $g \circ f$ is onto we must show that, for any $c \in C$, there is some $a \in A$ with $g(f(a)) = c$.) Because $g$ is onto, there is $b \in B$ with $g(b) = c$, and because $f$ is onto, there is $a \in A$ with $f(a) = b$. Then, for any $c \in C$, we have shown that there is an $a$ satisfying $g(f(a)) = g(b) = c$, as required.

**2.8** All functions involved map $\mathbb{R}$ to $\mathbb{R}$. (i) $x \mapsto 27x^3$, (ii) $x \mapsto (x-1)/3$, (iii) $x \mapsto \sqrt[3]{x} + 1$, (iv) $x \mapsto \sqrt[3]{x}/3$, (v) $x \mapsto \sqrt[3]{x}/3$. (It is not a coincidence that the last two are equal.)

# Chapter 3

# Counting

---

**Reading:** Ross & Wright: 5.1, 5.3, 5.4;
Chetwynd & Diggle: 4.1–4.6;
Grimaldi: 1.3, 1.4;
Grossman: 9.1–9.5.

---

Counting things is central to discrete mathematics, and has applications throughout computer science: to compute how much memory a program uses, or how long it takes to run, or indeed sometimes whether it will work at all. Some textbooks use the phrase **enumerative combinatorics** as a pretentious synonym for "counting".

The things we count are objects in sets, and the number of objects is the set's cardinality. The set is defined as all the objects with a certain **property**. Usually we begin by abstracting the problem into the familiar language of discrete maths. For example, the problem of counting the number of ways to distribute $n$ identical coins amongst $m$ people, such that each person gets at least one coin, is cast formally by associating a distribution of coins with an $m$-tuple indicating the number of coins given to each person. This effectively establishes a bijection between the problem and a set which we hope to be able to count (recall that bijections preserve cardinality). So in this example we would need to find $|\{(a_1, a_2, \ldots, a_m) \in \mathbb{N}^m \mid a_i \geq 1 \text{ for all } i, \text{ and } a_1 + a_2 + \cdots + a_m = n\}|$.

This chapter is devoted to techniques to help answer such questions, introduced by a series of examples. However, some simply-posed counting

questions can be very difficult to answer. The techniques covered here are sufficient to get started, and they only scratch the surface of a huge topic.

## 3.1 Laws of Sum and Product

In counting, as with so many other mathematical challenges, it is usually helpful to break down the problem into smaller parts. Then to combine the answers of counting subproblems we need some rules. There are three common **laws** here, although they are sometimes considered so obvious that they are not listed in textbooks. Since counting is about cardinality, it is not surprising that the laws derive from equations relating cardinality of sets.

The first counting law derives from the following fact: if $A$ and $B$ are disjoint finite sets then $|A \cup B| = |A| + |B|$.

**Law of Sum:** Let $P_1$ and $P_2$ be properties of objects which are **exclusive** (they cannot both be true of any object). The number of objects with *either* property $P_1$ *or* property $P_2$ is the number with property $P_1$ *plus* the number with property $P_2$.

This law extends to any number of properties as long as they are all pairwise exclusive (no two can happen at once). We will meet a more substantial generalisation, to non-exclusive properties, in Sect. 3.3.

**Example 3.1** How many positive integers less than a million have an odd number of digits?

**Answer** In the absence of any other instructions we should assume that the numbers are written in normal decimal notation, without leading zeros (i.e. 123 not 000123).

Positive integers less than a million have no more than 6 digits, so we need to find the number of positive integers with 1, 3, or 5 digits. These are exclusive properties because no integer has both $m$ and $n$ digits if $m \neq n$.

So how many positive integers have 1 digit? Clearly, the answer is 9.

How many have 3 digits? These are all the integers from 100 to 999, of which there are $999 - 100 + 1 = 900$ (the number of integers between $m$ and $n > m$, inclusive, is $n - m + 1$; $n - m$ is the number from $m$ to $n$ including only one of the two endpoints).

How many have 5 digits? These are all the integers from 10000 to 99999, of which there are $99999 - 10000 + 1 = 90000$.

Finally, by the sum law, there must be $9 + 900 + 90000 = 90909$ positive integers less than a million with an odd number of digits. •

The second law, which is really just a rearrangement of the first, derives from this fact about sets: if $A$ and $B$ are finite sets, with $B \subseteq A$, then $|A \setminus B| = |A| - |B|$.

**Law of Subtract:** Let $P_1$ and $P_2$ be properties, such that $P_1$ is true at least whenever $P_2$ is. Then the number of objects with property $P_1$ *but not* $P_2$ is the number with property $P_1$ *subtract* the number with property $P_2$.

Finally, the most important law relates to independent properties and derives from this fact: $|A \times B| = |A||B|$.

**Law of Product:** Suppose that we are counting the number of ways of making a sequence of choices, and the choices are **independent** in the sense that the number available at each stage does not depend on the choices made previously. Then the total number of ways of making the sequence of choices is the *product* of the number of choices at each stage.

Here is an example for which the solution combines the laws of subtract and product:

**Example 3.2** How many 6-digit positive integers contain at least one digit 7?

**Answer** There are $999999 - 100000 + 1 = 900000$ 6-digit numbers.

How many *do not* contain at least one digit 7? We use the product law: the first digit can be any one of 8 possibilities (anything but 0 or 7); the second through sixth digits can be any one of 9 (anything but 7). These are independent: whether the first digit is 0 or 7 does not affect our options for the second digit, and so on. So the product law applies and there are $8 \cdot 9^5 = 472392$ such integers.

Now apply the subtraction law: there are 900000 6-digit positive integers, of which 472392 have no 7s, so there must be $900000 - 472392 = 427608$ 6-digit positive integers with at least one 7. •

A very important example is the following.

**Example 3.3** In how many different orders can we arrange $n$ different objects?

**Answer** In the absence of any other instruction an **arrangement** refers to placing the objects in a sequence, without repetition.

There are $n$ choices for which object goes first. Having allocated that object, there are $n-1$ choices for which goes second, regardless of which object was selected first. Thus these are independent in the sense of the product law.

Then there are $n-2$ choices for which object goes third, and $n-3$ for the fourth. This pattern repeats until there are 2 choices for the penultimate object and only 1 choice when we place the final remaining object in the final place. Since the number of choices is in each case independent of previous choices, the total number of arrangements is $n(n-1)(n-2)\cdots 2\cdot 1$. •

This function is important enough to have its own name, which you are probably familiar with.

**Definition** The **factorial** function $(-)!$ is a function from $\mathbb{N}+$ to $\mathbb{N}+$ defined recursively by

$$1! = 1, \qquad (n+1)! = (n+1)(n!)$$

Unwinding the recursion, $n! = n(n-1)(n-2)\cdots 2\cdot 1$. It is common to extend the factorial function's domain to $\mathbb{N}$ by setting $0! = 1$.

(In fact, the factorial function can be generalised to all positive real numbers: $\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}\,\mathrm{d}t$ satisfies $\Gamma(n) = (n-1)!$ for $n \in \mathbb{N}_+$. This continuous function, known as the **Gamma function**, is defined for all positive reals, and indeed can be extended to almost every part of the complex plane. It is a fascinating function, but not within our discrete mathematics syllabus.)

## 3.2 The Technique of Double Counting

It can be easier to count the elements in a set twice, and then divide by two:

**Example 3.4** Each of $n$ teams in a league plays every other team once during a season. How many games must be played, in total, in a season?

**Answer** Let us number the teams $1, \ldots, n$. Fix a team $i$: they must play

each of $n-1$ teams (they do not play against themselves!); this applies to each of the $n$ teams. So, by the product law, it seems that there are $n(n-1)$ games played in total. But we have double counted, because if team $i$ plays against $j$ then it is not also necessary to count $j$ against $i$. Exactly twice too many games have been counted, so the correct answer is $\frac{n(n-1)}{2}$. •

Quite often we count a set more than twice over, as in this very important example:

**Example 3.5** In how many different ways can we select $k$ out of $n$ distinguishable objects, disregarding the order of selection?

**Answer** First we answer the question: in how many different ways can be select $k$ out of $n$ distinguishable objects, if the order of selection matters? We have seen similar questions before: there are $n$ choices for the first selection, $n-1$ for the second, and so on down to $n-k+1$ for the $k$-th selection. In total, $n(n-1)\cdots(n-k+1)$. This can be written compactly as $\frac{n!}{(n-k)!}$ and is known as the number of **permutations** of $k$ objects from $n$.

Now return to the original question. We have counted each selection $k!$ times, because there are $k!$ different ways to order the selection of $k$ objects. Therefore the number of **combinations** of $k$ objects from $n$ is $\frac{n!}{(n-k)!k!}$. •

For a concrete example, if we have to select 4 objects from 6, there are $\frac{6!}{2!} = 360$ ways to select the objects when the order matters, and we have over-counted by $4! = 24$ if the order is irrelevant, so overall there are $\frac{6!}{2!4!} = 15$ selections without regard to order.

The formula $\frac{n!}{(n-k)!k!}$ is so important that it has a special shorthand.

**Definition** The **combinatorial coefficients** (or **binomial coefficients**) $\binom{n}{k}$ are defined for all nonnegative integers $n$ and $k$ satisfying $0 \le k \le n$ by

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

Recall that, conventionally, $0! = 1$, so that $\binom{n}{0} = \binom{n}{n} = 1$. You can think of $\binom{\cdot}{\cdot}$ as a partial function from $\mathbb{N}^2$ to $\mathbb{N}_+$. It is not defined for $k > n$.

The combinatorial coefficients make up **Pascal's triangle** and they have all sorts of interesting properties but we will not dwell on them here. One

we will highlight is simply that $\binom{n}{k} = \binom{n}{n-k}$. This follows directly from the definition, or alternatively by noting that the number of ways to select $k$ from $n$ is exactly the same as the number of ways of *not* selecting $n-k$ from $n$.

In some books, $\binom{n}{k}$ is written ${}^nC_k$, and the related quantity $\frac{n!}{(n-k)!}$ written ${}^nP_k$.

So far we have used discrete mathematics techniques to help with counting. Here is an example of how counting can help develop results in discrete mathematics.

**Claim 3.6** For $n \in \mathbb{N}$,

$$2^n = \binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \cdots + \binom{n}{n}.$$

**Proof** Consider the set $\mathcal{P}(\{1, 2, \ldots, n\})$. Its members are all the subsets of $\{1, \ldots, n\}$, and we will count its cardinality in two different ways.

First, there are two choices for whether 1 occurs in a subset, two for whether 2 occurs, ..., and two for whether $n$ occurs. Everything is independent of everything else, so the product law tells us that there are $2^n$ different subsets.

Second, each subset of $\{1, \ldots, n\}$ has cardinality between 0 and $n$. How many subsets of $\{1, \ldots, n\}$ have cardinality $k$? This is just the number of ways of choosing $k$ from $n$, $\binom{n}{k}$. No subset can have two different cardinalities, so the sum law applies: the total number of subsets is the number with zero elements plus the number with one element, and so on. This is $\binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \cdots + \binom{n}{n}$. ■

The same result can be proved in a large number of different ways, but it illustrates the idea of counting the same object in two different ways in order to relate combinatorial formulae. This is an attractive and powerful part of discrete mathematics.

Now we have an example of a counting problem where a bit of thought is required before we can apply our combinatorial techniques. The following problem is about the distribution of *indistinguishable* objects into $k$ groups, but it turns out to be related to the problem we just studied, that of splitting *distinguishable* objects into 2 groups (taken/not taken) of given size.

**Example 3.7** How many ways can $n$ (indistinguishable) objects be placed in $k$ (distinguishable) boxes?
**Answer** This "distribution" problem can be related to the selection problem as follows. Let us write O for an object, and | for the dividing lines between the boxes. For example, one way to put 5 objects in 3 boxes is to put three in the first, none in the second, and two in the last: write this as "OOO | | OO". Or putting all five objects in the middle box is written " | OOOOO | ". With $n$ objects and $k$ boxes there must be $n$ O's and $k-1$ |'s. Each different way to distribute $n$ objects in $k$ boxes corresponds to a string of $n$ O's and $k-1$ |'s (formally, we have found a bijection between the different distributions and the strings).

How many such strings are there? There are $n+k-1$ positions in the string, and we must decide which $k-1$ are to be |'s (the others are O's). There are $\binom{n+k-1}{k-1}$ such choices. This is also equal to $\binom{n+k-1}{n}$. •

The result is important enough to be worth memorising. Here is a variation:

**Example 3.8** How many ways can $n$ (indistinguishable) objects be placed in $k$ boxes so that each box receives at least one object?
**Answer** If $k > n$ then there are not enough objects to go around, so the answer is zero. Otherwise the trick is to set aside $k$ objects, one for each box, and distribute the remaining $n-k$ without restriction. According to the formula derived in Example 3.7 there are $\binom{(n-k)+k-1}{k-1} = \binom{n-1}{k-1}$ such distributions. •

These answers can be applied to another problem, although the connection is not immediately obvious.

**Example 3.9** How many positive integers less than a million have their digits sum to 9?
**Answer** By padding a number with leading zeros (e.g. 123 becomes 000123) we can imagine that they are all exactly 6 digits long. Each such number with digits summing to 9 corresponds to a distribution of 9 objects into 6 boxes: the number of objects in box $i$ corresponding to digit $i$. So the number of 6-digit integers with digits summing to 9 is the same as the number of distributions of 9 indistinguishable objects into 6 boxes, which we have already determined is $\binom{9+6-1}{6-1} = \binom{14}{5} = \frac{14\cdot13\cdot12\cdot11\cdot10}{5\cdot4\cdot3\cdot2} = 2002$. •

Notice that the correspondence in the last example only works because there are fewer than 10 objects to distribute; the same question distributing more

than 9 objects is more difficult, because individual digits are not allowed to be greater than 9.

## 3.3 The Inclusion-Exclusion Principle

The inclusion-exclusion principle applies to counting cardinality of unions in the case when the sets are not disjoint (so that the sum law does not apply directly). It is easier to stick to terminology of set cardinality rather than talking about properties, because the combination of properties will become rather complicated.

The simplest case applies to cardinality of the union of two sets:

**Binary Inclusion-Exclusion:** For finite sets $A$ and $B$,

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

Why is this true? When we add $|A|$ and $|B|$, all the elements of $A \cap B$ are counted twice, so correcting for the double-counting yields the formula.

**Example 3.10** How many 6-digit positive integers contain both a digit 7 and a digit 9? (More than one occurrence of either or both is also allowed).

**Answer** We have already computed, in Example 3.2, that there are 427608 6-digit integers containing one or more 7's; call the set of these integers $A$. By symmetry, there are the same number containing one or more 9's; call the set of them $B$.

Rearranging the inclusion-exclusion principle, we have $|A \cap B| = |A| + |B| - |A \cup B|$ and the questions asks for the value of the left hand side, so we need to work out the number of 6 digit positive integers containing *either* a 7 and a 9. As in Example 3.2, we count the number which contain *neither* a 7 nor a 9: $7 \cdot 8^5 = 229376$ (7 choices for the first digit, 8 for the others). So there must be $900000 - 229376 = 670624$ with either a 7 or a 9.

Therefore there must be $427608 + 427608 - 670624 = 184592$ 6-digit numbers with both a 7 and a 9. •

Now we present the fully-general version of the principle.

**The Inclusion-Exclusion Principle:** For finite sets $A_1, \ldots, A_n$,

$$\begin{aligned} |A_1 \cup A_2 \cup \cdots \cup A_n| = \quad & \big(|A_1| + |A_2| + \cdots + |A_n|\big) \\ & - \big(|A_1 \cap A_2| + |A_1 \cap A_3| + \cdots + |A_{n-1} \cap A_n|\big) \\ & + \big(|A_1 \cap A_2 \cap A_3| + \cdots + |A_{n-2} \cap A_{n-1} \cap A_n|\big) \\ & - \cdots \\ & + (-1)^{n-1}|A_1 \cap A_2 \cap \cdots \cap A_n| \end{aligned}$$

The second row contains intersections of all distinct *pairs* of the $A_i$, the third all intersections of distinct *triplets*, and so on.

Why is this true? If we compute the sum $|A_1| + |A_2| + \cdots + |A_n|$ we have double-counted all the members of the two or more $A_i$, but subtracting$|A_1 \cap A_2| + |A_1 \cap A_3| + \cdots + |A_{n-1} \cap A_n|$ has subtracted too much: all those which are members of three or more $A_i$ must be added back in, and so on. The result can be proved using just the case $n = 2$, but we will not do so here.

Here is a classic application of the inclusion-exclusion principle.

**Example 3.11** In how many ways can we rearrange $n$ objects so that none stays in the same place?

**Answer** Permutations where every object moves are called **derangements**. (An equivalent formulation is to ask for permutations of $\{1, \ldots, n\}$ where each number $i$ is not in place $i$.) We will count the number of permutations in which at least one object does stay in the same place and use the subtraction law to complete the calculation.

Let us write $A_i$ for the set of permutations of $n$ objects where at least object $i$ is *not* moved. It is easy to see that, for any $i$, $|A_i| = (n-1)!$ because object $i$ is fixed in position $i$ and the others permuted without restriction. Similarly, $|A_i \cap A_j|$, for $i \neq j$, can be computed by fixing objects $i$ and $j$ and permuting the rest: $(n-2)!$ possibilities. And $|A_i \cap A_j \cap A_k| = (n-3)!$ as long as $i$, $j$, and $k$ are distinct. The pattern continues all the way to $|A_1 \cap \cdots \cap A_n| = (n-n)! = 1$.

Now, how many pairwise intersections $A_i \cap A_j$ are there? Because order of $i$ and $j$ is irrelevant, and $i \neq j$, there are $\binom{n}{2}$ such sets. How many triple intersections $A_i \cap A_j \cap A_k$? There must be $\binom{n}{3}$ because we are choosing three of the $A$'s out of $n$ without repetition or regard to order. Again, the pattern continues.

Now we are in a position to apply the inclusion-exclusion principle.

$$\begin{aligned}
&|A_1 \cup \cdots \cup A_n| \\
&\quad = \quad (|A_1| + \cdots + |A_n|) \\
&\qquad - (|A_1 \cap A_2| + \cdots + |A_{n-1} \cap A_n|) \\
&\qquad + \cdots \\
&\qquad + (-1)^{n-1}|A_1 \cap \cdots \cap A_n| \\
&\quad = \quad \tbinom{n}{1}(n-1)! - \tbinom{n}{2}(n-2)! + \tbinom{n}{3}(n-3)! - \cdots + (-1)^{n-1}\tbinom{n}{n}1 \\
&\quad = \quad \tfrac{n!}{1!} - \tfrac{n!}{2!} + \tfrac{n!}{3!} - \cdots + (-1)^{n-1}\tfrac{n!}{n!}
\end{aligned}$$

This is the number of permutations which are not derangements. The total number of permutations on $n$ elements is $n!$, so the number of derangements is $\frac{n!}{2!} - \frac{n!}{3!} + \cdots + (-1)^n \frac{n!}{n!}$. •

Another application of the inclusion-exclusion principle can be found in the following section.

## 3.4 Ceiling and Floor Functions

We would now like to address this problem: how many positive integers, less than or equal to $n$, are divisible by 3? The relevant multiples of 3 are $3, 5, 9, 12, \ldots, 3k$ such that $3k$ is the largest multiple of 3 less or equal to than $n$. Can we give a formula for $k$? If $n$ is itself a multiple of three then the answer is just $n/3$, but if $n$ is not then we need to take $n/3$ and round down. The operation of rounding down, or rounding up (which we sometimes also need) are useful enough functions to have special symbols.

**Definition** The **floor** function $\lfloor - \rfloor : \mathbb{R} \to \mathbb{Z}$ is given by $\lfloor x \rfloor = \max\{n \in \mathbb{Z} \mid n \leq x\}$; $\lfloor x \rfloor$ is the greatest integer no bigger than $x$.

The **ceiling** function $\lceil - \rceil : \mathbb{R} \to \mathbb{Z}$ is given by $\lceil x \rceil = \min\{n \in \mathbb{Z} \mid n \geq x\}$; $\lfloor x \rfloor$ is the least integer no smaller than $x$.

For example, $\lfloor 1.2 \rfloor = \lceil 0.8 \rceil = 1$, $\lfloor -1.2 \rfloor = \lceil -2.8 \rceil = 2$, and $\lfloor 3 \rfloor = \lceil 3 \rceil = 3$. Note that in some mathematical books, $\lfloor - \rfloor$ is written $[-]$ and there is no direct notation for the ceiling function.

Now we have a concise formula for the number of positive integers, less than or equal to $n$, which are divisible by 3: $\lfloor n/3 \rfloor$. More generally, the number

divisible by $k$ (where $k \in \mathbb{N}_+$) must be $\lfloor n/k \rfloor$. To count the number of positive integers between $m$ and $n$ which are divisible by $k$, use the subtraction law (see the practice questions).

Here is a final example.

**Example 3.12** How many positive integers, (strictly) less than 1000, are divisible by one or more of the first three prime numbers: 2, 3 or 5?
**Answer** Let us write $D_k$ for the set of positive integers, less than 1000, which are divisible by $k$. By the previous argument, $|D_k| = \lfloor \frac{999}{k} \rfloor$. We need to find $|D_2 \cup D_3 \cup D_5|$ and the inclusion-exclusion formula is applicable:

$$\begin{aligned}|D_2 \cup D_3 \cup D_5| =& |D_2| + |D_3| + |D_5| - |D_2 \cap D_3| - |D_2 \cap D_5| - |D_3 \cap D_5| \\ &+ |D_2 \cap D_3 \cap D_5|\end{aligned}$$

Next, observe that an integer is divisible by both 2 and 3 if and only if it is divisible by 6, similarly for the other pairs, and an integer is divisible by all of 2, 3, and 5 if and only if it is divisible by 30. So we have

$$\begin{aligned}|D_2 \cup D_3 \cup D_5| &= |D_2| + |D_3| + |D_5| - |D_6| - |D_{10}| - |D_{15}| + |D_{30}| \\ &= \lfloor \tfrac{999}{2} \rfloor + \lfloor \tfrac{999}{3} \rfloor + \lfloor \tfrac{999}{5} \rfloor - \lfloor \tfrac{999}{6} \rfloor - \lfloor \tfrac{999}{10} \rfloor - \lfloor \tfrac{999}{15} \rfloor + \lfloor \tfrac{999}{30} \rfloor \\ &= 499 + 333 + 199 - 166 - 99 - 66 + 33 = 733.\end{aligned}$$

•

## 3.5 Interesting Diversion: Multinomial Coefficients

The binomial coefficients count the number of ways of selecting $k$ objects from $n$ (distinguishable) objects. This can also be seen as the number of ways of splitting $n$ objects into two groups: one of size $k$ and one of size $n-k$. The **multinomial coefficients** count the number of ways of splitting $n$ objects into $g \geq 2$ groups of specified size.

**Claim 3.13** The number of ways in which $n$ objects can be split into $g$ groups, with $n_i$ in the group $i$, is

$$\frac{n!}{n_1! n_2! \cdots n_g!}$$

as long as $n_1 + n_2 + \cdots + n_g = n$.

**Proof** The condition $n_1 + n_2 + \cdots + n_g = n$ simply ensures that the sizes

of the groups adds up to the right total.

We use the product law. There are $\binom{n}{n_1}$ ways to choose $n_1$ objects for the first group, then we have $n - n_1$ left so $\binom{n-n_1}{n_2}$ ways to choose $n_2$ for the second group, then $\binom{n-n_1-n_2}{n_3}$ ways to choose $n_3$ for the third group, and so on down to $\binom{n-n_1-n_2-\cdots-n_{g-1}}{n_g}$ (which equals 1) for the final group. Writing out the binomial coefficients we have

$$\frac{n!}{n_1!(n-n_1)!}\ \frac{(n-n_1)!}{n_2!(n-n_1-n_2)!}\ \frac{(n-n_1-n_2)!}{n_3!(n-n_1-n_2-n_3)!}\ \cdots\ \frac{(n-n_1-\cdots-n_{g-1})!}{n_g!(n-n_1-n_2-\cdots-n_g)!}.$$

Cancelling the fractions and noting that $(n - n_1 - n_2 - \cdots - n_g)! = 0! = 1$ we are left with $\frac{n!}{n_1!n_2!\cdots n_g!}$. ■

By analogy with binomial coefficients, the multinomial coefficient $\frac{n!}{n_1!n_2!\cdots n_g!}$ is written

$$\binom{n}{n_1 n_2 \cdots n_g}.$$

**Example 3.14** If the card game *bridge*, all 52 cards in a pack are distributed amongst four players in a *deal*, with each player given 13 cards. How many different deals are there?

**Answer** We are asked how many ways there are to split 52 cards (all different) into 4 groups of 13. By the preceding argument, the number is $\binom{52}{13\ 13\ 13\ 13} = \frac{52!}{13!^4} = 53,644,737,765,488,792,839,237,440,000.$ •

## Practice Questions

**3.1** How many 6-digit positive integers have no repeated digits?

**3.2** We computed, in Example 3.2, that there are 427608 6-digit integers containing one or more $d$'s when $d$ is the digit 7. For which digit $d$ is the answer different, and what is the answer in that case?

**3.3** How many different arrangements are there of the letters in the word ANAGRAM?

**3.4** How many positive integers less than 1000 have their digits sum to 4? What is the largest such integer?

**3.5** How many positive integers less than 1000 have their digits sum to 10? Remember that individual digits can only be as large as 9.

**3.6** In how many ways can 5 red balls and 5 black balls be distributed into 5 boxes? There are no restrictions on the number or colour of balls in each box, but distributions count as different if the colours of the balls in any box are not the same.

**3.7** Find a formula, in terms of $m$, $n$, and $k$, for the number of integers strictly between $m$ and $n$ which are divisible by $k$.

**3.8** How many positive integers less than 1000 are:

(i) divisible by 4?

(ii) divisible by 6?

(iii) divisible by both 4 and 6?

(iv) divisible by either 4 or 6 (or both)?

(v) divisible by either 4 or 6 but not both?

## Answers to Chapter 3 Practice Questions

**3.1** $9 \cdot 9 \cdot 8 \cdot 7 \cdot 6 \cdot 5 = 136080$.

**3.2** It is not true for $d = 0$. In this case the answer is $900000 - 9^6 = 368559$ (each digit can be anything except zero).

**3.3** If the letter As were distinguishable there would simply be 7! arrangements; because there are three As, this over-counts by a factor of 3!. Therefore there are 840 different arrangements.

**3.4** We are dealing with 3-digit integers and the number summing to 4 is, as in Example 3.9, the same as the number of ways of distributing 4 objects into 3 boxes: $\binom{4+3-1}{3-1} = \binom{6}{2} = 15$.
To find the largest 3-digit integer with a digit sum of 4, note that the "hundreds" digit is most significant, so make it as large as possible: the answer is 400.

**3.5** We are dealing with 3-digit integers. For now, ignore the condition that individual digits can only be as large as 9. The number of 3-tuples of nonnegative numbers summing to 10 is, as in Example 3.9, the same as the number of ways of distributing 10 objects into 3 boxes: $\binom{10+3-1}{3-1} = \binom{12}{2} = 66$. Now exclude cases when a "digit" is equal to 10: there are exactly 3 such cases, because if one "digit" equals 10 then the rest must be zero. So there are 63 true 3-digit numbers whose digits sum to 10.

**3.6** Imagine that the reds and blacks are distributed separately. According to Example 3.7 there are $\binom{4+4-1}{4-1} = \binom{7}{3} = 35$ ways to distribute the reds. By symmetry there are 35 ways to distribute the blacks. Because the distribution of reds and blacks are independent there must be $35^2 = 1225$ ways in total.

**3.7** $\left\lfloor \frac{n-1}{k} \right\rfloor - \left\lfloor \frac{m}{k} \right\rfloor$.

**3.8** (i) $\left\lfloor \frac{999}{4} \right\rfloor = 249$; (ii) $\left\lfloor \frac{999}{6} \right\rfloor = 166$; (iii) $\left\lfloor \frac{999}{12} \right\rfloor = 83$ (NB: this is equivalent to divisibility by 12, not 24); (iv) $249 + 166 - 83 = 332$; (v) $249 + 166 - 83 - 83 = 249$.

# Chapter 4

# Relations

---

**Reading:** Ross & Wright: 3.1, 3.4, parts of 11.4 & 11.5;
Chetwynd & Diggle: 3.1, 3.2, 3.4;
Grimaldi: parts of 5.1, 7.1, parts of 7.2, 7.4;
Grossman: 5.5.

---

Sets, functions and relations are mathematical concepts, but they also form the building blocks for formal specifications of computing systems. Like functions, relations associate elements of sets, but they do not have the same strict conditions; they can be viewed as functions with potentially multiple outputs, or no output at all, corresponding to each input. In the case of computing systems depending on unobserved features, such as the behaviour of networks, relations are necessary to model their behaviour. The applications, though, are for another day. We will confine ourselves to the mathematical presentation of relations, and their graphical description.

## 4.1 Definition

The definition of a relation is extremely straightforward.

**Definition** A **relation** $R$ on a set $A$ is a subset of $A \times A$.

A simple example is $R = \{(n, n+1) \mid n \in \mathbb{N}\}$. This is a relation on $\mathbb{N}$ (it is also a relation on any superset of $\mathbb{N}$). Another example is, for any set

$A$, $R = \{(a, a) \mid a \in A\}$: this is a relation on the set $A$. A more random example would be $R = \{(1, 1), (1, 3), (2, 1)\}$, a relation on $\{1, 2, 3\}$.

However, relations usually have a special **infix** notation. Instead of $(a, b) \in R$, we write $a\,R\,b$. And instead of $(a, b) \notin R$, $a \not R\, b$. This makes a lot more sense when you realise that familiar concepts like "$\leq$" are relations on sets of numbers. Other familiar relations over sets of numbers are "$=$" (elements are only related to themselves), "$\neq$" (elements are related to everything *except* themselves), "$<$", "$>$", and so on.

Returning to the example $R = \{(n, n + 1) \mid n \in \mathbb{N}\}$, which could also be defined by saying $n\,R\,n + 1$ for each $n \in \mathbb{N}$ (thus unambiguously listing all the pairs which are related, with the implication that all other pairs are not related), we have $0\,R\,1$ and $3\,R\,4$, but $0 \not R\, 0$, $0 \not R\, 2$ and $4 \not R\, 3$.

A common computer science example is found in the modelling of a system's behaviour. Suppose that the state of a program (the values of all its memory locations and variables) is encoded in a set $S$ (very likely $S$ is a big cartesian product). The relation $s_1\,R\,s_2$ might mean that it is possible for state $s_1$ to change, in one clock cycle, to $s_2$.

An important note: some books, and indeed some computer science courses given in Oxford, allow a relation to be "from a set $A$ to a set $B$", where $B$ is not necessarily the same set as $A$. Such a relation is a subset of $A \times B$; most of the concepts of this chapter can be transferred this setting. You will notice that the functions from $A$ to $B$ can be considered relations from $A$ to $B$ satisfying an additional condition: for each $a \in A$, there is exactly one $b \in B$ with $a\,R\,b$; in this sense, relations are a more general object then functions. More generally still, relations can be defined between three or more sets (for example a relation on $A$, $B$ and $C$ is a subset of $A \times B \times C$). In that case, our type of relation is called a **binary relation**, but we will not use these extensions and we will not need to state that the relations are binary.

## 4.2 Properties of Relations

The concept of relation is very general, allowing any subset of $A \times A$. Usually we are interested in relations with particular properties. Commonly-required properties include:

**Definition** A relation $R$ on a set $A$ is

- **reflexive** if $a\,R\,a$ for all $a \in A$;
- **symmetric** if $a\,R\,b$ implies $b\,R\,a$, for all $a, b \in A$;
- **antisymmetric** if $a\,R\,b$ and $b\,R\,a$ together imply $a = b$, for all $a, b \in A$;
- **transitive** if $a\,R\,b$ and $b\,R\,c$ together imply $a\,R\,c$, for all $a, b, c \in A$.

Let us look at some familiar relations on $\mathbb{N}$. "$\leq$" is reflexive, because $n \leq n$ for all $n \in \mathbb{N}$. It is antisymmetric, because if $m \leq n$ and $n \leq m$ then $m = n$. It is also transitive, because whenever $m \leq n$ and $n \leq p$ we know that $m \leq p$. But it is not symmetric: to demonstrate this we must give a concrete counterexample of where the symmetry rule fails, and $1 \leq 2$ will do since we do not also have $2 \leq 1$.

An important relation, usually defined on the set $\mathbb{N}_+$, is the *divides relation* written $|$. We say $m \,|\, n$ if $n$ is a whole multiple of $m$: we have $2 \,|\, 4$ and $3 \,|\, 9$, $1 \,|\, n$ for any $n \in \mathbb{N}_+$, but $4 \not| \, 2$ and $4 \not| \, 5$. This relation is reflexive (every number is a multiple of itself), antisymmetric (it requires some thought to prove this), transitive (the heart of the proof is that $n = km$ and $p = ln$ imply $p = klm$), but not symmetric ($1 \,|\, 2$ but $2 \not| \, 1$). The divides relation can be extended to $\mathbb{Z}$, but care must be taken about the zero cases.

The relation "=", defined on any nonempty set, has all of the four properties listed above. It has become common to use symmetrical symbols ($=$, $\neq$, $\approx$) for symmetric relations, and asymmetrical symbols ($<$, $\rightarrow$) for relations which are not symmetric, although there are always some exceptions, (including $|$).

Other properties of relations include:

**Definition** A relation $R$ on a set $A$ is

- **irreflexive** if $a\,R\,a$ is not true for *any* $a \in A$;
- **serial** if for every $a \in A$ there is a $b \in A$ satisfying $a\,R\,b$.

Do not confuse the property of being irreflexive with being not reflexive. To be irreflexive, a relation $R$ must have $a \not R\, a$ for *every* $a \in A$, but to be not reflexive it only needs $a \not R\, a$ for *some* $a \in A$.

Another common confusion is between being antisymmetric and being not symmetric. If $a$ and $b$ are distinct elements of the domain, symmetry says

that none or both of $a\,R\,b$ and $b\,R\,a$ is true; antisymmetry says that none or one of $a\,R\,b$ and $b\,R\,a$ is true. It is possible for a relation to be neither symmetric nor antisymmetric (can you find an example?), or both symmetric and antisymmetric ("=" is the most obvious example).

## 4.3 Equivalence Relations

Of particular importance are relations with the following property.

**Definition** A relation on $A$ is called an **equivalence relation** if it is

- reflexive,
- symmetric, and
- transitive.

There are many examples: on any set the equality relation "="; for a universe $\mathcal{U}$ the relation $A \sim B$ when $|A| = |B|$; on the set $\mathbb{Z}$ the relation $m \equiv n$ if $m$ and $n$ have the same parity (i.e. $2 \mid (m - n)$). In computer science, we might define a relation on *programs* (in some fixed language) by $P_1 \approx P_2$ if the same inputs to the programs always produce the same outputs. This is an important example which, when specified precisely, is called **observational equivalence**; it does not say that $P_1$ and $P_2$ have the same internal working, or are equally efficient, though.

When we are given an equivalence relation on $A$ we often want to know all the elements of $A$ which are related to some fixed $a$.

**Definition** If $\sim$ is an equivalence relation on $A$, and $a \in A$, we define

$$[a] = \{a' \in A \mid a' \sim a\}.$$

The sets $[a]$, for each $a$, are called the **equivalence classes** of $\sim$.

Sometimes, to avoid ambiguity, we might write $[a]_\sim$ if it is not clear what equivalence relation is meant.

In the examples above: in the first, the equivalence class $[n]_=$ is just $\{n\}$; in the second, the equivalence class $[A]_\sim$ is the set of all subsets of $\mathcal{U}$ with the same cardinality as $A$; in the third, the equivalence class $[n]_\equiv$ is the set of

all integers with the same parity of $n$, so $[0] = \{\ldots, -4, -2, 0, 2, 4, \ldots\}$ and $[1] = \{\ldots, -3, -1, 1, 3, \ldots\}$. You will notice that the equivalence classes do not overlap unless they are equal.

**Definition** A **partition** of a set $A$ is a collection of subsets $\{B_i \mid i \in I\}$ with all $B_i \subseteq A$, satisfying

(i) $\bigcup_{i \in I} B_i = A$,

(ii) $B_i \cap B_j = \emptyset$ for $i \neq j$ (i.e. the subsets are pairwise disjoint), and

(iii) $B_i \neq \emptyset$, for each $i \in I$ (although some definitions do not require this).

($I$ is some indexing set: it is almost equivalent to write a partition as $\{B_1, B_2, \ldots\}$).

In other words, a partition divides the set $A$ into disjoint subsets. There is a close connection between partitions of $A$ and equivalence relations on $A$: each equivalence relation determines a partition, and vice versa. Formally,

**Claim 4.1** For any set $A$,

(i) The equivalence classes of any equivalence relation form a partition of $A$;

(ii) Any partition of $A$ defines an equivalence relation;

(iii) Different equivalence relations correspond to different partitions.

**Proof** (i) Let $\sim$ be an arbitrary equivalence relation on $A$. We need to show that the union of the equivalence classes of $\sim$ are nonempty and cover all of $A$ (easy: $a \in [a]$) and that they are disjoint. Suppose that two equivalence classes are not disjoint, say $x \in [a] \cap [b]$. Then for any $y \in [a]$ we have

$$\begin{array}{ll} y \sim a & \text{by definition of } [a], \text{ given } y \in [a] \\ x \sim a & \text{by definition of } [a], \text{ given } x \in [a] \\ a \sim x & \text{by symmetry, and previous step} \\ y \sim x & \text{by transitivity} \\ x \sim b & \text{by definition of } [b], \text{ given } x \in [b] \\ y \sim b & \text{by transitivity} \end{array}$$

so $y \in [b]$. (Once familiar with the use of symmetry and transitivity, it is not necessary to write out all the steps.) We have shown $[a] \subseteq [b]$. Repeating

the argument with $a$ and $b$ swapped, we have $[b] \subseteq [a]$. So if equivalence classes $[a]$ and $[b]$ overlap at all, they must be equal.

(ii) Given a partition $\mathcal{P} = \{B_i \mid i \in I\}$, define an equivalence relation by $a \sim b$ if $a$ and $b$ are both members of any of the $B_i$. It is easy to see that this relation is reflexive, symmetric, and transitive.

(iii) If $\sim_1$ and $\sim_2$ are different relations then there is a pair with $a \sim_1 b$ but $a \not\sim_2 b$ (or vice versa). Then the partition corresponding to $\sim_1$ does have $a, b$ in the same component, but the partition corresponding to $\sim_2$ does not. ■

We have established that there is a **bijection** between the set of equivalence relations on $A$ and the set of partitions of $A$, without constructing the bijection explicitly.

## 4.4 Operations on Relations

There are operations on relations, analogous to the operations of inverse and composition on functions.

**Definition** If $R$ is a relation on $A$ then the **converse relation** $R^{-1}$ is defined by

$$a\, R^{-1} b \quad \text{if} \quad b\, R\, a.$$

If $R$ and $S$ are relations on $A$ then the **composition** relation $S \circ R$ is defined by

$$a\, (S \circ R)\, b \text{ if there is some } x \in A \text{ such that } a\, R\, x \text{ and } x\, S\, b.$$

Beware! The relational converse has many different notations. I have seen $R^c$, $R^{\leftarrow}$ and $R^o$ all used for what we write $R^{-1}$. It is also sometimes called the **inverse** relation, but we reserve that word for functions. As with functions, $S \circ R$ is sometimes written $R; S$.

Take, for example, the relation on $\mathbb{N}$ given by $n\, R\, (n+1)$. The converse relation $R^{-1}$ given by $(n+1)\, R^{-1} n$. Note that 0 is not related to anything by the converse. The composition $R \circ R$ is given by $n (R \circ R) (n+2)$. Or take the not-equal relation $\neq$ (this example works on any set with more than two elements): since it is symmetric, its converse relation is also $\neq$. The

composition $\neq \circ \neq$ relates all pairs because, given elements $a$ and $b$, we can choose any other element $c$: we have $a \neq c$ and $c \neq b$ therefore $a(\neq \circ \neq)b$.

Observe that the composition of a relation $R$ with itself is like "doing $R$ twice". We often want to reason about the iterated effect of a relation, leading to the following definition.

**Definition** If $R$ is a relation on $A$ then the **transitive closure** $R^+$ is defined by

$$a\,R^+\,b \text{ if there is some } x_0, x_1, \ldots, x_n \in A \text{ with } n \geq 1 \text{ such that}$$
$$a = x_0, \quad x_0\,R\,x_1, \quad x_1\,R\,x_2, \quad \ldots, \quad x_{n-1}\,R\,x_n, \quad x_n = b.$$

If $R$ is a relation on $A$ then the **reflexive transitive closure** $R^*$ is defined by

$$a\,R^*\,b \text{ if there is some } x_0, x_1, \ldots, x_n \in A \text{ with } n \geq 0 \text{ such that}$$
$$a = x_0, \quad x_0\,R\,x_1, \quad x_1\,R\,x_2, \quad \ldots, \quad x_{n-1}\,R\,x_n, \quad x_n = b.$$

That is, there is a chain of elements of $A$, with each adjacent pair related by $R$, from $a$ to $b$: for $a\,R^+\,b$ the chain must have at least $R$-step in it; for $a\,R^*\,b$ there can be none (i.e. it includes the case $a = b$).

The transitive and reflexive transitive closure are so-called because they have the following property: $R^+$ is the smallest relation, containing $R$, that is transitive, and $R^*$ is the smallest relation, containing $R$, that is reflexive and transitive. Here "smallest" means with respect to inclusion: $R^+$, as a set of pairs, is a subset of every transitive relation containing $R$, etc. A relation $R$ which is already transitive satisfies $R^+ = R$, and a reflexive transitive relation $R$ satisfies $R^* = R$.

As an example, imagine the relation on $\mathbb{N}^2$ defined by $(m, n+1)\,R\,(m+1, n)$ for $m, n \in \mathbb{N}$. We have the chain $(3,0)\,R\,(2,1)\,R\,(1,2)\,R\,(0,3)$, so $(3,0)\,R^+\,(0,3)$. We do not have $(3,0)\,R^+\,(3,0)$, but we do have $(3,0)\,R^*\,(3,0)$.

Transitive closure is an important concept in computer science because, for example, if $R$ relates states before and after one step of computation, $R^*$ relates an initial state to any possible resulting state.

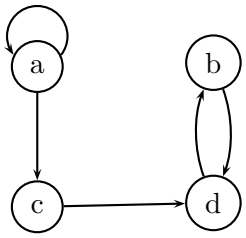
## 4.5 Drawing Relations

It is easy to define regular relations like "$<$" or $n\,R\,(n+1)$, but tedious to define a nonstandard relation by listing all its pairs. It is difficult to understand a relation such as $R = \{(a,a),(a,c),(b,d),(c,d),(d,b)\}$, a relation on $\{a,b,c,d\}$, and awkward to compute compositions or transitive closures from the uninformative listing of its members. However it is attractive to *draw* relations on small sets as a **digraph**.

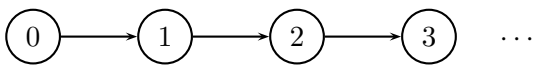
**Definition** A **directed graph** (**digraph**) consists of a set of **nodes** $N$ and a set of **edges** $E \subseteq N \times N$. We say that there is an edge (or **arrow**) from $n_1$ to $n_2$ if $(n_1, n_2) \in E$. Digraphs are depicted by drawing the nodes, as labelled points in the plane, and an arrow from $n_1$ to $n_2$ whenever $(n_1, n_2) \in E$.

You will notice that this is the same as the definition of a relation on $N$, but written in different terminology! (In computer science it often happens that the same concept is given equivalent formulations by different names, in different terminology, in various topics.) So this suggests a way to draw relations on $A$, with a node for each element of $A$ and an arrow from $a$ to $b$ if $a\,R\,b$.

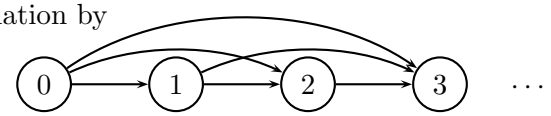
The relation $R$, defined above, can be described concisely by the following picture.




The plus-1 relation $n\,R\,(n+1)$ can be drawn as

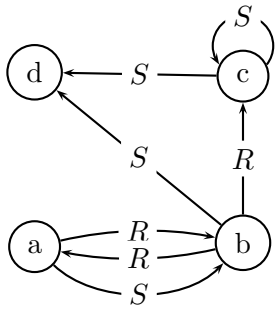



and the "$<$" relation by



This representation is particularly intuitive when the relation concerned relates something "before" to something "after", for example when it describes possible next-step behaviour of a computer program. Furthermore, it is easy to see, from the diagram, whether a relation is reflexive (all nodes must have a loop: an edge directly to itself), symmetric (all arrows have a partner going in the opposite direction) or antisymmetric (no arrow has a partner going in the opposite direction). It is not so easy to see whether a relation is transitive, however.

Indeed, common operations on relations also have nice versions expressed graphically: taking the converse of a relation reverses the direction of all arrows, and the transitive closure corresponds to multiple-step paths (i.e. $a\,R^+\,b$ if there is path through the diagram, starting at $a$ and following arrows to $b$). If $R$ and $S$ are relations on the same set we can draw them on the same graph, labelling edges indicating $R$ or $S$ (such a structure is called a **labelled digraph**). This can make it easy to see $S \circ R$. For example, if $R = \{(a, b), (b, a), (b, c)\}$ and $S = \{(a, b), (b, d), (c, c), (c, d)\}$ are relations on $\{a, b, c, d\}$ we draw them as




and can easily compute $S \circ R = \{(a, d), (b, b), (b, c), (b, d)\}$.

There is less need to draw equivalence relations: since we know that they correspond to partitions of the set it is easier to draw or list the corresponding partition rather than all the edges.

## 4.6 Interesting Diversion: Counting Relations

If $|A| = n$, how many different relations are there on $A$? How many are reflexive/symmetric/transitive? How many are equivalence relations? Some

of these questions are quite simple, and some are difficult.

Counting all the relations on $A$ is straightforward: remember that any subset of $A \times A$ is a relation, so the number of relations is $|\mathcal{P}(A \times A)| = 2^{n^2}$. Another way to prove the same fact is to imagine a relation as a table, one row and column for each member of $A$, and a tick in the row for $a$ and column for $b$ if $a$ is related to $b$: there are $n^2$ entries in the table, and each can be a tick or a cross without restriction. Two choices repeated independently $n^2$ times gives $2^{n^2}$ choices.

Let us count the number of symmetric relations on $A$. The symmetry condition requires that, for $a \neq b$, either $a\,R\,b$ and $b\,R\,a$, or $a \not R\, b$ and $b \not R\, a$, but it does not impose any condition on whether $a\,R\,a$. In our table of $n^2$ ticks or crosses, then, there is no restriction on the diagonal elements, but their must be reflectional symmetry about the leading diagonal. We have a free choice of the $n$ elements on the diagonal and the $n(n-1)/2$ elements below the diagonal, but then the elements above the diagonal are fixed by the symmetry property. Therefore there are $n(n+1)/2$ independent choices of tick or cross, giving $2^{\frac{n(n+1)}{2}}$ symmetric relations in total.

Counting reflexive relations, or antisymmetric relations, is easy using same techniques. Counting transitive relations is difficult: there is no closed formula, although the number of transitive relations on small sets can be counted by hand. The number of equivalence relations on $A$ is, of course, equal to the number of partitions, and we will find a way to count them in the next chapter.

## Practice Questions

**4.1** For each of the following relations, determine whether they are (a) reflexive, (b) symmetric, (c) anti-symmetric, (d) transitive, (e) irreflexive, (f) serial.

(i) On $\mathbb{N}$, $a \, R \, b$ if $a \leq b$.

(ii) On $\mathbb{N}$, $a \, R \, b$ if $a > b$.

(iii) On $\mathbb{N}$, $a \, R \, b$ if $a + b$ is even.

(iv) On the set of words in the English language, $w_1 \, R \, w_2$ if at least one letter occurs in both words $w_1$ and $w_2$.

**4.2** Which of the following are equivalence relations?

(i) On $\mathbb{N}$, $a \, R \, b$ if $a \leq b$.

(ii) On $\mathbb{N}$, $a \, R \, b$ if $a \neq b$.

(iii) On the set of words in the English language, $w_1 \, R \, w_2$ if $w_1$ is an anagram of $w_2$ (including the case when $w_1 = w_2$).

**4.3** Determine all the equivalence classes of the equivalence relation $\sim$ on $\mathbb{Z}$ defined by $a \sim b$ if either $3|a$ and $3|b$, or $3 \nmid a$ and $3 \nmid b$.

**4.4** List all the partitions of the set $\{1, 2, 3\}$.

**4.5** Find a relation on $\mathbb{N}$ which is irreflexive, antisymmetric, and serial. Then draw the graph of a relation on the set $\{a, b, c\}$ with the same properties.

**4.6** Let $A = \{a, b, c, d\}$ and $R$ be the relation $\{(a, b), (b, c), (c, d), (d, a)\}$. Compute $R \circ R$, $R \circ (R^{-1})$ and $R^+$. (It is probably helpful to draw $R$.)

**4.7** What familiar relations on $\mathbb{Z}$ are the converses of "$<$", "$\geq$", and "$=$"?

**4.8** If $|A| = n$, how many *reflexive* relations are there on $A$?

## Answers to Chapter 4 Practice Questions

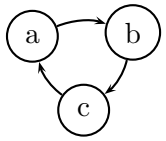
**4.1** (i) reflexive, anti-symmetric, transitive, serial;
(ii) anti-symmetric (yes: the condition is vacuously true), transitive, irreflexive;
(iii) reflexive, symmetric, transitive, serial;
(iv) reflexive, symmetric, serial.

**4.2** Only (iii).

**4.3** $\{\ldots, -6, -3, 0, 3, 6, \ldots\}$ and $\{\ldots, -5, -4, -2, -1, 1, 2, 4, 5, \ldots\}$.

**4.4** $\{\{1,2,3\}\}$, $\{\{1,2\},\{3\}\}$, $\{\{2,3\},\{1\}\}$, $\{\{1,3\},\{2\}\}$, $\{\{1\},\{2\},\{3\}\}$.

**4.5** "$<$" will do. On $\{a, b, c\}$, one example is




(there is only one other example, the converse of the depicted relation).

**4.6** $\{(a,c),(c,a),(b,d),(d,b)\}$; $\{(a,a),(b,b),(c,c),(d,d)\}$; $A \times A$.

**4.7** "$>$", "$\leq$", and "$=$".

**4.8** $2^{n(n-1)}$.

# Chapter 5

# Sequences

---

**Reading:** Ross & Wright: 4.2, 4.4, 4.5, 4.6;
Chetwynd & Diggle: not covered;
Grimaldi: 4.1, 4.2;
Grossman: 7.1–7.4.

---

Sequences, particularly sequences of numbers, occur in many computer science disciplines. Often, the sequence is specified by a recursive formula, and later "solved" to find a closed form for the $n$-th term of the sequence. In this chapter we will explore some examples of sequences, and the important proof technique called **induction**. Induction is often used to prove that a closed form is the correct solution, but finding the closed form in the first place can be harder.

## 5.1 Definition

Sequences are usually infinite, and we write them as $(x_1, x_2, \ldots)$, but they are best formalized as functions:

**Definition** A **sequence** is a function whose domain is $\mathbb{N}$ (or $\mathbb{N}_+$). The argument of the function is usually written as a subscript, e.g. $f_n$ instead of $f(n)$. The entire sequence is denoted $(f_n)$.

Sequences often have letters like $(x_n)$ or $(a_n)$; when the sequence refers to the **time complexity** of a program (the number of steps it takes to run) then it is usually $(t_n)$ or, abandoning the subscripted argument, $T(n)$.

The obvious way to define a sequence is to give the function explicitly. For example, $x_n = 2n$ defines the sequence $(2, 4, 6, 8, \ldots)$ or $(0, 2, 4, 6, \ldots)$ (whether a sequence starts at 0 or 1, or even some other number, is a matter of convenience). Or $a_i = i^2$ gives the sequence $(0, 1, 4, 9, 16, \ldots)$. But sequences are often defined **recursively**, by writing $x_n$ as a formula of some of the previous $x_i$ and then giving enough of the first few values of the sequence for the rest to be determined.

Recursively-defined sequences include $x_0 = 0$, $x_{n+1} = x_n + 2$ (this defines the same sequence as $x_n = 2n$) and $f_0 = 1$, $f_{n+1} = (n + 1)f_n$ (this is the sequence of factorials) but $x_n$ is allowed to depend on more than just the immediately preceding sequence term. An important example is

**Example 5.1** The **Fibonacci sequence** or (Fibonacci numbers) is the sequence given by

$$F_0 = 0, \quad F_1 = 1, \quad F_{n+2} = F_{n+1} + F_n.$$

The first few terms of the Fibonacci sequence are $0, 1, 1, 2, 3, 5, 8, 13, 21, 34, \ldots$

**Example 5.2** The sequence $x_n$ defined by $x_0 = 0$, and $x_n = x_{\lfloor \frac{n}{2} \rfloor} + 1$ for $n > 0$, begins $(0, 1, 2, 2, 3, 3, 3, 3, 4, \ldots)$.

It is important to include cases to get the recursion started (they are called **boundary conditions**): the sequences generated by $y_{n+1} = 2y_n$, $y_0 = 0$ and $y'_{n+1} = 2y'_n$, $y'_0 = 1$ are completely different, and the Fibonacci recurrence $F_{n+2} = F_{n+1} + F_n$ does not define a sequence unless we specify the first two terms.

A sequence defined recursively is also known as a **recurrence relation**. If we want to find a nonrecursive formula for the $n$-th term of a recursively-defined sequence, we speak of **solving** the recurrence relation.

In fact, it is possible to solve the preceding recurrence relations. The $n$-th Fibonacci number $F_n$ has the closed form

$$F_n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n$$

and the sequence $x_n$, above, has

$$x_n = \lfloor \log_2 n \rfloor + 1$$

for $n \geq 1$. You can probably see where the latter comes from, but the former might seem rather mysterious (indeed it is not even obvious that the formula gives a whole number!). Methods to solve recurrences of the Fibonacci type are briefly covered in Section 5.5.

Note that the same sequence might be specified by any number of different recurrences. For example, you can check that $x_n = n!$ is generated by either $x_0 = 1$, $x_{n+1} = (n+1)x_n$ or by $x_0 = 1$, $x_1 = 1$, $x_{n+2} = (n+1)(x_{n+1} + x_n)$.

## 5.2 Proof by Induction

In many areas of mathematics and computer science we are asked to prove that some statement, which depends on a natural number $n$, is true for all $n$. Sometimes we can do so directly, but sometimes the proof depends so much on the value of $n$ that we seem forced to perform infinitely many different proofs, one for each $n$. In such a situation, the method of induction can be helpful.

**Principle of Induction:** Let $S(n)$ be a statement involving the natural number $n$. If we prove $S(0)$, and we prove that $S(k+1)$ is true whenever $S(k)$ is true, then this is sufficient to prove that $S(n)$ for all $n \in \mathbb{N}$.

When performing an induction we call the part which proves $S(0)$ the **base case** and the part which, assuming $S(k)$, proves $S(k+1)$ the **inductive step**. In the latter, the assumption that $S(k)$ is true is known as the **inductive hypothesis** (often abbreviated as IH). It should be easy to understand why a proof by induction is correct: it shows that $S(0)$ is true, so $S(1)$ is true, so $S(2)$ is true, and eventually each $S(n)$ is proved true. The principle holds because $0 \in \mathbb{N}$, and $k \in \mathbb{N}$ implies $k+1 \in \mathbb{N}$ is really the *definition* of the set of natural numbers.

As an example of a proof by induction, we will prove a fact about a sequence.

**Example 5.3** Recall the Fibonacci numbers: $F_0 = 0$, $F_1 = 1$, $F_{n+2} = F_{n+1} + F_n$. We can prove that $2 \mid F_{3n}$ for each $n \in \mathbb{N}$ by induction on $n$.

First, the base case $n = 0$. We need to prove that $2 \mid F_0$: $F_0 = 0$, and 0 is a multiple of 2, so we are done.

Second, the inductive step. Suppose that we have already proved that $2 \mid F_{3k}$: we need to prove that $2 \mid F_{3k+3}$. Using the recurrence relation which defined $F_n$, we calculate

$$\begin{aligned} F_{3k+3} &= F_{3k+2} + F_{3k+1} \\ &= 2F_{3k+1} + F_{3k} \\ &= 3F_{3k} + 2F_{3k-1} \end{aligned}$$

But remember we have assumed that $F_{3k}$ is even, and $2F_{3k-1}$ is certainly even, so we have proved that $2 \mid F_{3k+3}$. That completes the proof by induction. ■

If we are to prove that something holds for $n \geq 1$ (or more generally $n \geq m$) we merely modify the base case to be $n = 1$ (or $n = m$). In fact, other proofs by induction are possible: if we prove $S(0)$ and $S(1)$, and that $S(k)$ implies $S(k + 2)$, then we have covered all of $\mathbb{N}$. More complex alternatives are possible, but rare in practice when trying to prove a result for all $n \in \mathbb{N}$. It is also possible to use induction to prove that something is true for members of other sets, and then the structure of the set determines the structure of the inductive proof. We will not see examples of these "structural" inductions in this course, but they are very common in Functional Programming.

One slightly modified type of induction we will need is sometimes called *strong induction*.

**Principle of Strong Induction:** Let $S(n)$ be a statement involving the natural number $n$. If we prove $S(0)$, and we prove that $S(k + 1)$ is true assuming that $S(j)$ is true for *all* $j \leq k$, then this is sufficient to prove that $S(n)$ for all $n \in \mathbb{N}$.

An example of a strong induction is:

**Claim 5.4** Every $n \in \mathbb{N}_+$ can be written as the sum of distinct Fibonacci numbers.

**Proof** The base case is $n = 1$, which is just a single Fibonacci number ($F_1$); nothing to prove.

For the inductive step, we may assume that each of $1, 2, 3, \ldots, k$ can all be written as the sum of distinct Fibonacci numbers and we must prove that

the same is true of $k+1$. If $k+1$ is already a Fibonacci number, we are finished. If not, $k+1$ lies between two consecutive Fibonacci numbers $F_j$ and $F_{j+1}$ (the Fibonacci sequence increases indefinitely), i.e. $F_j < k+1 < F_{j+1}$. Write $k+1 = F_j + (k+1-F_j)$. Since $k+1-F_j \leq k$, the inductive hypothesis tells us that $k+1-F_j$ can be written as the sum of distinct Fibonacci numbers. Now remember that $k+1 < F_{j+1} = F_j + F_{j-1}$ (using the Fibonacci recurrence) so $k+1-F_j < F_{j-1}$. This means that none of the distinct Fibonacci numbers which sum to $k+1-F_j$ could possibly be $F_j$, or the sum would be too big. Thus we can write $k+1$ as a sum of distinct Fibonacci numbers: $F_j$ plus all those which sum to $k+1-F_j$. This completes the proof by induction. ■

A proof technique closely related to induction is called the **minimal counterexample**. It is a form of proof by contradiction. If we want to prove that something is true for all $n$, we suppose (for a contradiction) that it is not true. Then we examine the smallest $n$ for which it is false, trying to deduce that, in fact, it must be false for a smaller $n$ as well. That is a contradiction. This is best illustrated by a simple example

**Claim 5.5** The recurrence $x_1 = 1$, $x_2 = 3$, $x_{n+2} = 4x_{n+1} + 3x_n$ generates a sequence of odd numbers.

**Proof** This can easily be proved by induction, but even quicker is to use the minimal counterexample technique. Suppose that $x_m$ is the first even number in the sequence. $m$ cannot be 1 or 2, so $x_m = 4x_{m-1} + 3x_{m-2}$. This implies that $x_{m-2}$ is even, a contradiction with the assumption that $x_m$ was the first even number in the sequence. Therefore there cannot be *any* even numbers in the sequence. ■

## 5.3 Sigma Notation and Sums of Sequences

First, some standard mathematical notation.

**Definition** If $(a_n)$ is a sequence then we can write $\sum_{i=m}^{n} a_i$ as a shorthand for the sum of the segment $a_m + a_{m+1} + \cdots + a_n$. Analogously, the notation $\prod_{i=m}^{n} a_i$ is the product $a_m \cdot a_{m+1} \cdots a_n$.

This notation can be extended to $\sum_{i=1}^{\infty} a_i$, which performs the infinite summation, but analysis of infinite sums or products (whose values are not necessarily defined) is beyond the scope of this course.

For example, we have $\sum_{i=1}^{4} i^2 = 1^2 + 2^2 + 3^2 + 4^2 = 30$, and $\prod_{i=2}^{4}(i+1) = 3 \cdot 4 \cdot 5 = 60$. Given a sequence $a_i$, we can generate a new sequence called the **partial sums** of $a_i$ defined by $s_n = \sum_{i=1}^{n} a_i$ and, using induction, we can sometimes give a simple formula for the partial sums of a sequence. For example, the sum of the first $n$ square integers:

**Claim 5.6** For $n \in \mathbb{N}_+$, $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$.

**Proof** (The result is also true for $n = 0$ if we adopt the standard convention that a sum of no terms equals zero.)

This is a classic example of proof by induction. For the base case ($n = 1$) the left side is 1 and the right side is 6/6.

For the inductive step, suppose the result is true for $n = k$ and look at $n = k + 1$:

$$\begin{array}{rcll} \sum_{i=1}^{k+1} i^2 & = & \sum_{i=1}^{k} i^2 + (k+1)^2 & \text{extracting the final term} \\ & = & \frac{k(k+1)(2k+1)}{6} + (k+1)^2 & \text{by the inductive hypothesis} \\ & = & \frac{(k+1)(k(2k+1)+6(k+1))}{6} & \text{rearranging} \\ & = & \frac{(k+1)(k+2)(2k+3)}{6} & \text{factorising} \end{array}$$

and this is exactly the statement for $n = k + 1$. That completes the proof by induction. ■

This demonstrates that proofs by induction often have the flavour of verifying a formula, rather than finding it: we could not have attempted the proof without knowing, or guessing, the value of the sum.

## 5.4 Sequences Associated with Counting

Recurrence relations arise in counting problems, and sometimes it is possible to use some ingenious reasoning to find a recursive formula for counting some set when direct methods fail.

We have already met the **derangements**, in Chapter 3. Instead of the combinatorial formula derived there, the number of derangements of $n$ objects $d_n$ can be described by a recurrence relation.

**Claim 5.7** The sequence $d_n$ satisfies

$$d_1 = 0, \quad d_2 = 1, \quad d_n = (n-1)(d_{n-1} + d_{n-2}) \text{ for } n \geq 2.$$

**Proof** That $d_1 = 0$ and $d_2 = 1$ is immediate from the definition of the derangements. We must then argue that $d_n = (n-1)(d_{n-1} + d_{n-2})$.

Consider a derangement on objects $\{1, 2, \ldots, n\}$, and suppose that object $n$ is found in place $i$: there are $n-1$ possibilities for $i$ (object $n$ cannot be in place $n$). Now if we *swap* object $n$ and object $i$ there are two exclusive cases:

- After swapping $n$ and $i$, the first $n-1$ objects still form a derangement.
- After swapping $n$ and $i$, the first $n-1$ objects do not form a derangement.

There are $d_{n-1}$ ways in which the former can happen and in the latter, because we supposed that we began with a derangement on all $n$ objects, the only way we can fail to have a derangement is if we now have object $i$ in place $i$ leaving $d_{n-2}$ derangements of the other elements. In total, then, $(n-1)(d_{n-1} + d_{n-2})$ derangements of all $n$ objects. ■

It is interesting to note that the recurrence $d_n = (n-1)(d_{n-1} + d_{n-2})$ is also satisfied by the factorial function (which counts the total number of permutations, whether derangements or not) if the boundary condition is altered.

Another sequence associated with counting is

**Definition** The number of partitions of a set with cardinality $n$ is written $B_n$. The sequence $B_0, B_1, \ldots$ is known as the **Bell numbers** (after the sci-fi author and mathematician Eric Temple Bell) and begins $1, 1, 2, 5, 13. \ldots$.

It is difficult to find a closed form for $B_n$, but we can give a recurrence relation which enables the sequence to be computed step-by-step.

**Claim 5.8** The Bell numbers are generated by: $B_0 = 1$,

$$B_{n+1} = \sum_{i=0}^{n} \binom{n}{i} B_i.$$

**Proof** There is only one set with zero cardinality ($\emptyset$) and only one partition

of it.

Now consider a set of cardinality $n+1$, say $A = \{a_1, \ldots, a_{n+1}\}$. Consider what happens to $a_{n+1}$ in a partition of $A$:

- $a_{n+1}$ could be a singleton set in the partition. In that case there are $B_n$ ways to partition the other $n$ elements, to which $\{a_{n+1}\}$ is added to make a partition of $A$.
- $a_{n+1}$ could be in a set with exactly one other element $a_i$, in the partition. There are $n = \binom{n}{1}$ choices for this other element, and for any such choice there are $B_{n-1}$ ways to partition the other $n-1$ elements of $A$, to which $\{a_i, a_{n+1}\}$ is added to make a partition of $A$.
- $a_{n+1}$ could be in a set with exactly two other elements $a_i, a_j$, in the partition. There are $\binom{n}{2}$ choices for these other elements, and for any such choice there are $B_{n-2}$ ways to partition the other $n-2$ elements of $A$, to which $\{a_i, a_j, a_{n+1}\}$ is added to make a partition of $A$.
- ...
- $a_{n+1}$ could be in a set with exactly $n-1$ other elements in the partition. There are $\binom{n}{n-1}$ choices for these elements, and for any such choice there are $B_1$ ways to partition the other element of $A$.
- $a_{n+1}$ could be in a set with all $n$ other elements in the partition. There is $1 = \binom{n}{n}$ choice.

These cases are disjoint, so we can add up all the choices giving $\sum_{i=0}^{n} \binom{n}{i} B_i$ different partitions of $A$. (Here we used $\binom{n}{k} = \binom{n}{n-k}$.) ■

The mathematician Dobinski proved, in the 1870s, that the Bell numbers can be given by the *infinite* sum

$$B_n = \frac{1}{e} \sum_{k=0}^{\infty} \frac{k^n}{k!}.$$

Even though an infinite sum seems a rather pointless "solution", this formula draws an interesting connection between Bell numbers and the Poisson distribution found in probability.

You might be able to convince yourself that Dobinski's formula is correct by substituting this formula into the right hand side of the recurrence

$\sum_{i=0}^{n} \binom{n}{i} B_i$ and rearranging. If you use the **binomial theorem**

$$(a+b)^n = \sum_{i=0}^{n} \binom{n}{i} a^i b^{n-i}$$

you can reduce to $\frac{1}{e}\sum_{k=0}^{\infty} \frac{k^{n+1}}{k!}$, suggesting that the Dobinski formula does satisfy the required recurrence. But infinite sums cannot always be manipulated in the same way as finite sums, and you need some more mathematics, not covered in this course, to make the calculation into a proper proof.

## 5.5 Interesting Diversion: Solving Linear Recurrence Relations

One class of recurrence relations is both common and reasonably easy to solve. These are the constant-coefficient linear recurrence relations. There now follows a quick primer on how to solve them; the reasoning behind the method will not be included. You may find these methods useful in the probability course.

First, the **homogeneous** constant-coefficient linear recurrences. They have the form

$$\lambda_m x_n + \lambda_{m-1} x_{n-1} + \cdots + \lambda_0 x_{n-m} = 0, \tag{5.1}$$

where all the $\lambda_i$ are real numbers, plus some boundary conditions (usually $m$ are required.) Examples include $x_{n+1} = 3x_n$ (which can be rewritten in the form $x_n - 3x_{n-1} = 0$), $x_n - x_{n-1} - x_{n-2} = 0$ (the Fibonacci recurrence), and $x_n + 3x_{n-1} + 3x_{n-2} + x_{n-3} = 0$.

To solve these recurrences, try a solution of the form $x_n = r^n$, where $r$ is a constant. Substituting into (5.1) and dividing through by $r^{m-n}$ we have

$$\lambda_m r^m + \lambda_{m-1} r^{m-1} + \cdots + \lambda_0 = 0. \tag{5.2}$$

This is called the **characteristic polynomial**. Note that *any* value of $r$ satisfying (5.2) will give a solution of (5.1).

Find all the roots $r_1, r_2, \ldots, r_m$ of the characteristic polynomial (including complex roots, if there are any). As long as all these roots are distinct, all solutions of (5.1) will be of the form $x_n = A_1 r_1^n + A_2 r_2^n + \cdots + A_m r_m^n$,

where the $A$'s are constants. The boundary conditions of the recurrence will determine the $A$'s. If some of the roots are repeated, add extra multiples of $n$ to the repeat factors until they are all distinct. For example, if the characteristic polynomial factorizes as $(r-2)^2(r-3)^3$ then the solutions to (5.1) will be of the form $x_n = A_1 2^n + A_2 n 2^n + A_3 3^n + A_4 n 3^n + A_5 n^2 3^n$.

**Example 5.9** Consider the linear homogeneous recurrence

$$x_n - 4x_{n-1} + 5x_{n-2} - 2x_{n-3} = 0, \quad x_0 = 0, \quad x_1 = 1, \quad x_2 = 2.$$

The characteristic polynomial is $r^3 - 4r^2 + 5r - 2 = 0$, which factorizes as $(r-1)^2(r-2)$. Therefore the general solution of the recurrence, without any boundary conditions, is

$$x_n = A + Bn + C2^n$$

(because 1 is a repeated root of the characteristic polynomial we need the terms $A1^n$ and $Bn1^n$). Now substitute into the boundary conditions: we have $0 = A + C$, $1 = A + B + 2C$, $2 = A + 2B + 4C$. Solving these three simultaneous equations gives $A = 0$, $B = 1$, $C = 0$, so in fact the solution to the original recurrence relation is just $x_n = n$.

We might want to solve an **inhomogeneous** constant-coefficient linear recurrence. This is of the form

$$\lambda_m x_n + \lambda_{m-1} x_{n-1} + \cdots + \lambda_0 x_{n-m} = f(n), \tag{5.3}$$

plus some boundary conditions (again usually $m$ of them). Note that if $y_n$ is a solution of the homogeneous recurrence (without the term $f(n)$) and $z_n$ is a solution of the inhomogeneous recurrence, then $x_n + z_n$ is also a solution of the inhomogeneous recurrence.

We begin by "guessing" a solution to the inhomogeneous recurrence, without regard to the boundary conditions. Experience shows that when $f(n)$ is a polynomial of degree $n$, we should guess a polynomial of degree $n$, and if $f(n) = a^n$ we should guess a multiple of $a^n$, and you can find other rules of thumb in textbooks. Then solve the homogeneous recurrence, without regard to boundary conditions. Finally, add together the two solutions, and use the boundary conditions to determine any missing constants.

**Example 5.10** Consider the linear inhomogeneous recurrence

$$x_n - 2x_{n-1} = 3^n + 5n, \quad x_0 = 0.$$

Because of the form of the inhomogeneity, we "guess" that $x_n = A3^n + Bn + C$ will find a solution. Substituting into the recurrence, we obtain the equation

$$A3^n + Bn + C - 2A3^{n-1} - 2B(n-1) - 2C = 3^n + 5n$$

and equating coefficients gives: $(3^n)$ $A = 3$, $(n)$ $B = -5$, $(1)$ $C = -10$, so we have found the solution $z_n = 3^{n+1} - 5n - 10$. But this does not yet satisfy the boundary condition.

We solve the homogeneous recurrence $y_n - 2y_{n-1} = 0$, which is straightforwardly given by $y_n = D2^n$ for any $D$. Then we try $x_n = y_n + z_n = D2^n + 3^{n+1} - 5n - 10$ against the boundary condition, finding that $D = 7$. We have solved the original recurrence by

$$x_n = 7 \cdot 2^n + 3^{n+1} - 5n - 10.$$

Finally, we mention in passing that there is another technique for solving recurrences called **generating functions**. They are a powerful tool for certain types of recurrence, including nonlinear recurrences where the above techniques do not apply, but they do not appear in this course.

---

## Practice Questions

**5.1** Find recurrence relations which generate the sequences

(i) $(1, 4, 16, 64, 256, \ldots)$;

(ii) $(1, 4, 9, 16, 25, 36, 49, \ldots)$;

(iii) $x_n = \lfloor \log_3 n \rfloor$ (for $n \geq 1$).

**5.2** Find, using the techniques of Section 5.5, or simply by guessing the solution and proving by induction, the solution to the recurrence $u_0 = 0$, $u_1 = 1$, $u_{n+2} = 2u_{n+1} - u_n + 1$.

**5.3** Show, by induction on $n$, that $x_n = 2^n - 1$ solves the recurrence $x_1 = 1$, $x_{n+1} = 2x_n + 1$.

**5.4** Prove, by induction on $n$, that $\frac{1}{1\cdot 4} + \frac{1}{4\cdot 7} + \cdots + \frac{1}{(3n-2)(3n+1)} = \frac{n}{3n+1}$ for $n \geq 1$.

**5.5** Using the recurrence relation for the Bell numbers, compute $B_5$.

**5.6** If the recurrence relation for the number of derangements, $d_n = (n-1)(d_{n-1} + d_{n-2})$, $d_1 = 0$, $d_2 = 1$, is to be extended to $d_0$, what must be the value of $d_0$ for the recurrence still to hold? Does this make sense?

**5.7** Find the solutions of

(i) $x_0 = 1$, $x_{n+1} = 3x_n$;

(ii) $x_1 = 1$, $x_2 = 1$, $x_{n+2} = 3x_{n+1} - 2x_n$;

(iii) $x_0 = 0$, $x_1 = 0$, $x_2 = 0$, $x_{n+3} = 16x_{n+2} + 15x_{n+1} + 14x_n$.

(They do not require proof.)

**5.8** Using the techniques in Section 5.5, find the solution of $x_1 = 1$, $x_2 = 3$, $x_{n+2} = 3x_{n+1} - 2x_n$

## Answers to Chapter 5 Practice Questions

**5.1** There are many answers, but the simplest are (i) $x_0 = 1$, $x_{n+1} = 4x_n$; (ii) $x_0 = 1$, $x_{n+1} = x_n + 2n + 1$; (iii) $x_1 = 0$, $x_2 = 0$, $x_n = x_{\lfloor n/3 \rfloor} + 1$ for $n \geq 3$.

**5.2** $u_n = n(n+1)/2$.

**5.3** The base case is $n = 1$: $2^1 - 1 = 1 = x_1$. Suppose the result for $n = k$. Then the right-hand side of the recurrence $2x_k + 1$, substituting the IH, equals $2(2^k - 1) + 1$, which equals $2^{k+1} - 1$ as required. That completes the induction.

**5.4** The base case is $n = 1$: the left hand side is $\frac{1}{4}$ and the right hand side is $\frac{1}{3 \cdot 1 + 1}$. For the inductive step, suppose that $\frac{1}{1 \cdot 4} + \frac{1}{4 \cdot 7} + \cdots + \frac{1}{(3k-2)(3k+1)} = \frac{k}{3k+1}$. Now compute $\frac{1}{1 \cdot 4} + \frac{1}{4 \cdot 7} + \cdots + \frac{1}{(3k-2)(3k+1)} + \frac{1}{(3k+1)(3k+4)} = \frac{k}{3k+1} + \frac{1}{(3k+1)(3k+4)}$. Simplifying, this equals $\frac{k(3k+4)+1}{(3k+1)(3k+4)} = \frac{3k^2+4k+1}{(3k+1)(3k+4)} = \frac{(3k+1)(k+1)}{(3k+1)(3k+4)} = \frac{k+1}{3(k+1)+1}$. This is the required result for $k + 1$. That completes the proof by induction.

**5.5** $B_5 = \binom{4}{0}1 + \binom{4}{1}1 + \binom{4}{2}2 + \binom{4}{3}5 + \binom{4}{4}15 = 52$.

**5.6** $d_0 = 1$. This makes sense because the only partition of the empty set does not leave element $i$ in place $i$ (there is no element $i$ to leave in place $i$!).

**5.7** (i) $x_n = 3^n$; (ii) $x_n = 1$; (iii) $x_n = 0$.

**5.8** $x_n = 2^{n+1} - 1$.

# Chapter 6

# Modular Arithmetic

---

**Reading:** Ross & Wright: 3.5, 4.7, 5.5;
Chetwynd & Diggle: only 3.9 (mostly not covered);
Grimaldi: 4.4, 5.5, 14.3;
Grossman: 12.1–12.4.

---

When arithmetic is performed on the remainder, after division by a fixed number $n$, this is known as **modular arithmetic**, and $n$ is called the **modulus**. This topic is usually considered part of pure mathematics (the branch known as **number theory**) and this chapter is the most mathematical of the course. But modular arithmetic has applications in computer science algorithms, particularly for computing with very large numbers, testing whether a number is prime (vital for encryption), and coding for reliable information transmission over a noisy network.

## 6.1 Definitions

We begin with a relation on $\mathbb{Z}$. Fix $n \in \mathbb{N}_+$, then write $x \equiv y \pmod{n}$ if $n \mid (x-y)$. This is equivalent to saying that $x$ and $y$ have the same remainder when divided by $n$. It is easy to see that this is an equivalence relation: $x$ certainly has the same remainder as $x$ (reflexivity); if $x$ and $y$ have the same remainder, so do $y$ and $x$ (symmetry); if $x$ and $y$ have the same remainder, and so do $y$ and $z$, then $x$ and $z$ have the same remainder (transitivity).

The equivalence classes are easy to compute: any $x$ is related to $x-2n$, $x-n$, $x$, $x+n$, $x+2n$, etc, so the $n$ equivalence classes are $[0], [1], \ldots, [n-1]$.

Now for some arithmetic.

**Claim 6.1** If $x_1 \equiv x_2 \pmod{n}$ and $y_1 \equiv y_2 \pmod{n}$, then

(i) $x_1 + y_1 \equiv x_2 + y_2 \pmod{n}$, and

(ii) $x_1 y_1 \equiv x_2 y_2 \pmod{n}$.

**Proof** We have $n \mid (x_1 - x_2)$ so $x_1 - x_2 = kn$ for $k \in \mathbb{Z}$, and $n \mid (y_1 - y_2)$ so $y_1 - y_2 = ln$ for $l \in \mathbb{Z}$.

(i) Adding, we deduce $x_1+y_1-x_2-y_2 = (k+l)n$, so $n \mid \big((x_1+y_1)-(x_2+y_2)\big)$.

(ii) Multiplying by $y_1$ and then again by $x_2$, we have $x_1y_1 - x_2y_2 = (x_1 - x_2)y_1 + x_2(y_1 - x_2) = (ky_1 + x_2l)n$ so $n \mid (x_1y_1 - x_2y_2)$. ■

This allows us to define the operations of addition and multiplication **modulo** $n$, on the *equivalence classes*: $[x] + [y] = [x + y]$ and $[x] \cdot [y] = [xy]$. We have proved that the answers do not depend on which members of $[x]$ and $[y]$ are chosen to represent the class. Equivalently, we have defined addition and multiplication binary operators on the set $\mathbb{Z}_n$.

This is **modular arithmetic** (mod $n$), in which multiples of $n$ are discarded and only the remainders kept. An example of a calculation in modular arithmetic is

$$13 \cdot 15 + 17 \equiv 4 \cdot 6 + 8 \equiv 32 \equiv 5 \pmod{9}.$$

In modular arithmetic calculations we can replace any number by another with the same remainder after division by the modulus. Usually we remove all multiples of the modulus (this is called **reduction** (mod $n$)) but sometimes it is convenient reduce numbers to negative remainders. For example, an equivalent calculation is

$$13 \cdot 15 + 17 \equiv (4 \cdot -3) - 1 \equiv -13 \equiv 5 \pmod{9}.$$

Negative numbers (and subtraction) do exist in modular arithmetic; because $n - x \equiv -x \mod n$, every number does have a negative (mathematicians call $-x$ an **additive inverse**). The situation is not so simple for division, as we shall see later.

## 6.2 Exponentiation

Having seen that the result of addition and multiplication (mod $n$) require only the remainders of the arguments, it is natural to assume that the same applies to exponentiation. It does not, quite.

**Claim 6.2** If $x_1 \equiv x_2 \pmod{n}$ and $y \in \mathbb{Z}$ then

(i) $x_1^y \equiv x_2^y \pmod{n}$, but

(ii) $y^{x_1} \equiv y^{x_2} \pmod{n}$ need not hold.

**Proof** (i) By $x_1 \equiv x_2 \pmod{n}$ and the rule for modular multiplication we have $x_1^2 \equiv x_2^2 \pmod{n}$, so $x_1^3 \equiv x_2^3 \pmod{n}$, and so on. (It is also true for $x_1^0 = 1 = x_2^0$). This proof can be formalised by induction.

(ii) The simplest counterexample is $0^0 = 1$ and $0^2 = 0$ with $n = 2$. (It is correct that $0^0 = 1$, but there are many other counterexamples if this one looks like a technicality.) ■

Therefore we can say that $13^{12} \equiv 3^{12} \pmod{10}$, but it is not true that $13^{12} \equiv 13^2 \pmod{10}$ (check for yourself). In fact, there are different laws for exponentiation, which are not very hard to prove but beyond the scope of this course. When $p$ is a prime number, we can guarantee $y^{x_1} \equiv y^{x_2} \pmod{p}$ as long as $x_1 \equiv x_2 \pmod{p-1}$; it seems a bit strange that one equation (mod $p$) is connected to another (mod $p-1$), until you get used to it! This law, and its extensions, are the heart of the well-known public key encryption scheme **RSA**.

If we have to compute a large exponentiation, for example $x^{600} \pmod{11}$, we can reduce $x$ mod $n$, but would like to avoid computing the 600th power to find the remainder. This is possible using the **method of repeated squaring**, which amounts to the recurrence

$$x^y = \begin{cases} 1 & \text{if } y = 0 \\ x \cdot (x^2)^{\frac{y-1}{2}} & \text{if } y \text{ is odd} \\ (x^2)^{\frac{y}{2}} & \text{if } y > 0 \text{ is even} \end{cases}$$

which is true for integers $x$ and $y \geq 0$. So to compute $2^{600} \pmod{11}$, we can apply the recurrence repeatedly and reduce the base of the exponent at

each stage:

$$\begin{array}{rcll} 2^{600} & \equiv & (2^2)^{300} & \equiv & 4^{300} & \pmod{11} \\ & \equiv & (4^2)^{150} & \equiv & 5^{150} & \pmod{11} \\ & \equiv & (5^2)^{75} & \equiv & 3^{75} & \pmod{11} \\ & \equiv & 3 \cdot (3^2)^{37} & \equiv & 3 \cdot 9^{37} & \pmod{11} \\ & \equiv & 3 \cdot 9 \cdot (9^2)^{18} & \equiv & 5 \cdot 4^{18} & \pmod{11} \\ & \equiv & 5 \cdot (4^2)^9 & \equiv & 5^{10} & \pmod{11} \\ & \equiv & (5^2)^5 & \equiv & 3^5 & \pmod{11} \\ & \equiv & 3 \cdot (3^2)^2 & \equiv & 3 \cdot 9^2 & \pmod{11} \\ & \equiv & 3 \cdot 4 & \equiv & 1 & \pmod{11}. \end{array}$$

We end with a little observation about squares (mod 4). Since we know that the remainder of $x \pmod 4$ determines the remainder of $x^2 \pmod 4$, consider all the possibilities:

$$0^2 \equiv 0 \pmod 4, \quad 1^2 \equiv 1 \pmod 4, \quad 2^2 \equiv 0 \pmod 4, \quad 3^2 \equiv 1 \pmod 4.$$

So always $x^2 \equiv 0$ or $1 \pmod 4$: a square number can never have remainder 2 or 3 when divided by 4.

## 6.3 MOD and DIV

We briefly consider two binary operators on (positive) integers, which are related to modular arithmetic.

**Definition** $\text{MOD} : \mathbb{N} \times \mathbb{N}_+ \to \mathbb{N}$ and $\text{DIV} : \mathbb{N} \times \mathbb{N}_+ \to \mathbb{N}$ are written infix and defined by:

$$\begin{array}{rcl} m \text{ MOD } n & = & \text{the remainder when } m \text{ is divided by } n \\ m \text{ DIV } n & = & \lfloor m/n \rfloor \end{array}$$

More usefully, we have the equation $m = n \cdot (m \text{ DIV } n) + m \text{ MOD } n$, expressing $m$ uniquely as an integer multiple of $n$ plus a remainder term.

We have introduced these functions because most programming languages provide them as primitives (MOD is often the symbol "%"). Many languages attempt to extend the domains to $\mathbb{Z}^2$, but unfortunately it is not clear what

the answers should be when the arguments are negative (particularly the modulus): the meaning of 3 MOD $(-2)$ is not obvious. Different languages may implement different functionality, so it is sensible to exercise caution when using DIV and MOD operators, whatever they are called in your programming language of choice, on signed numbers.

## 6.4 Euclid's Algorithm and Multiplicative Inverses

Recall the relation $m \mid n$ ($m$ divides $n$). Whether a number divides another is at the heart of modular arithmetic, and it is often important to find the *largest* integer which divides a set of numbers.

**Definition** For $m, n \in \mathbb{N}_+$ the **greatest common divisor (gcd)** of $m$ and $n$, written $\gcd(m, n)$, is the greatest integer $g$ which satisfies both $g \mid m$ and $g \mid n$.

If $\gcd(m, n) = 1$ then $m$ and $n$ are said to be **coprime** (in some books, **relatively prime**).

The definition extends to any set of positive integers: $\gcd(m_1, m_2, \ldots, m_n)$ is the greatest integer $g$ for which $g \mid m_i$ for all $i$, and a set is called coprime if its gcd is 1.

(An equivalent definition, sometimes more convenient, is that $\gcd(m, n) = g$ if i) $g \mid m$, ii) $g \mid n$, and iii) $l \mid m$ and $l \mid n$ together imply $l \mid g$.)

For example, $\gcd(12, 10) = 2$, $\gcd(9, 3) = 3$, $\gcd(5, 14) = 1$ (5 and 14 are coprime).

A few notes about divisors. Recall that an integer $p > 1$ is called a **prime number** if its only divisors are 1 and $p$ (i.e. $m \mid p$ implies $m = 1$ or $m = p$). The number 1 is *not* considered to be a prime number. Now if $n < p$ is another positive integer, we must have $\gcd(n, p) = 1$: $n$ and $p$ cannot have any common divisors greater than 1 if $p$ does not have any other divisors.

Let us consider how to *compute* the greatest common divisor of a pair of integers. It is rather hard work to find all the divisors of $m$ and $n$, and then look for the greatest, if $m$ and $n$ are large numbers. And, when we want to compute gcds in practice (e.g. for cryptography purposes), they often

are very large numbers indeed. Thankfully there is a very efficient method, which works because of the equation $\gcd(m, n) = \gcd(m, n - m)$:

**Euclid's Algorithm** for finding $\gcd(m, n)$. Set $a = m$, $b = n$. Write $a = qb + r$, with $0 \leq r < b$ (i.e. $q = a$ DIV $b$ and $r = a$ MOD $b$). If $r = 0$, then $\gcd(m, n) = b$. Otherwise set $a = b$ and $b = r$, and repeat.

When using Euclid's Algorithm in pen-and-paper calculations, it is sensible (and often useful, see below) to write each line $a = qb+r$ out in full. Here is an example of Euclid's Algorithm, to compute $\gcd(1029, 273)$. Initially we have $a = 1029$, $b = 273$, then:

$$\begin{array}{lrcll} (1) & 1029 & = & 3 \cdot 273 + 210 & (\text{now } a = 273 \text{ and } b = 210) \\ (2) & 273 & = & 1 \cdot 210 + 63 & (\text{now } a = 210 \text{ and } b = 63) \\ (3) & 210 & = & 3 \cdot 63 + 21 & (\text{now } a = 63 \text{ and } b = 21) \\ (4) & 63 & = & 3 \cdot 21 + 0 & \end{array}$$

Then we finish, having computed $\gcd(1029, 273) = 21$.

Now Euclid's Algorithm can be used for another purpose: by reversing the calculations, we can always express $\gcd(m, n)$ as an integer multiple of $m$ plus an integer multiple of $n$. We do this by beginning at the second-last line: (3) gives

$$21 = 1 \cdot 210 - 3 \cdot 63$$

then substitute (2) into this to get

$$21 = 1 \cdot 210 - 3 \cdot (273 - 210) = 4 \cdot 210 - 3 \cdot 273$$

then substitute (1) to get

$$21 = 4 \cdot (1029 - 3 \cdot 273) - 3 \cdot 273 = 4 \cdot 1029 - 15 \cdot 273.$$

If asked to implement the above procedure on a computer, this can be done attractively using matrices, but we will not consider the implementation of Euclid's algorithm in this course. The procedure ensures that we can always express $\gcd(m, n)$ as a sum of integer multiples of $m$ and $n$ (this fact is sometimes called **Euclid's Theorem**). This has practical importance – in cryptography we sometimes need to solve $mx+ny = \gcd(m, n)$ for integer $x$ and $y$ – as well as theoretical consequences.

One of the latter is

**Claim 6.3** If $p$ is prime then for each nonzero $x \in \mathbb{Z}_p$ there exists $y \in \mathbb{Z}_p$ such that $xy \equiv 1 \pmod{p}$.

This follows because $\gcd(x, p) = 1$ when $1 \leq x < p$. We have shown that every nonzero number has a **multiplicative inverse** (mod $p$), as long as $p$ is prime. It is not true when the modulus is composite: try to solve $2x \equiv 1 \pmod{6}$. When the modulus is fixed, the multiplicative inverse of $x$ is often written $x^{-1}$.

## 6.5 The Pigeonhole Principle

The Pigeonhole Principle (also known by other names including the Box Principle or the Drawer Principle) is a delightfully simple and surprisingly powerful proof technique. When stated it seems almost trivial:

**The Pigeonhole Principle** You cannot put more than $n$ pigeons into $n$ pigeonholes without having at least two pigeons in one hole.

In practice, we most often use the principle in the following way: we make a set of $n$ numbers (pigeons), we prove that they can take at most $m$ different values where $m < n$ (pigeonholes), and then deduce that two numbers in the set must be equal. As a simple example,

**Claim 6.4** In any group of people there are at least two with the same number of friends within the group.

**Proof** Suppose that the group is $n$ people. The wording of the question implies that $n \geq 2$ and we may assume that friendship is a symmetric relation(!) and discount the possibility that someone is friends with themselves. Let us write integers $f_1, f_2, \ldots f_n$ for the number of friends of each person within the group. We know that $0 \leq f_i \leq n - 1$, since each person has at most $n - 1$ potential friends.

We cannot immediately apply the pigeonhole principle to $\{f_1, \ldots, f_n\}$, because we have $n$ numbers and they might have $n$ different values. But it is impossible to have both $f_i = 0$ and $f_j = n-1$ for some $i$ and $j$, for $j$ cannot be friends with everybody if $i$ is friends with nobody.

For this reason, the $n$ numbers $\{f_1, \ldots, f_n\}$ can take at most $n-1$ different values. By the pigeonhole principle at least two of the numbers must be equal. ■

When attempting a proof in which the idea is that something "does not fit" inside something else, the pigeonhole principle is usually the key.

A computer science application of the pigeonhole principle now follows. A "data compression algorithm" is a method for shrinking computer files, usually by looking for redundancy in the file; examples include `gzip` or `rar`. Unless otherwise specified, the operation must be **lossless**, i.e. invertible: there must be a way to decompress and recover the original exactly. We are not considering so-called lossy compressors like `jpg` or `mp3`.

**Claim 6.5** No (lossless) data compression algorithm can reduce the size of any of its inputs, unless it increases the size of some of its inputs.

**Proof** Suppose, for a contradiction, that there is a data compression algorithm which reduces the size of some inputs, and perhaps preserves the size of some inputs, but never increases the input size. Let us measure input and output sizes in bits, for convenience.

Let $n$ be the length of the shortest input which compresses to a shorter output; suppose that the corresponding output is $m$ bits. Because $m < n$, and under the supposition that $n$ was the shortest input which was shrunk, all the inputs of length $m$ give outputs of length $m$.

But there are only $2^m$ possible outputs of length $m$, and all the inputs of length $m$ ($2^m$ of them) plus at least one input of length $n$ (the one which shrunk) are assumed to give outputs of length $m$: a total of at least $2^m + 1$ inputs giving up to $2^m$ different outputs. By the pigeonhole principle, at least two different inputs must correspond to the same output. This is a contradiction, because it is impossible to decompress that output to recover the original (which original should be returned?). ■

This is not a disaster for the science of data compression. It does not matter if `gzip` compression slightly increases the size of very unusual files, as long as it generally shrinks the sorts of files which are commonly stored on computers. This is indeed the case.

Here is another application of the pigeonhole principle.

**Claim 6.6** Take any set $A$ of $n+1$ distinct positive integers less than or equal to $2n$, where $n \in \mathbb{N}_+$. Then

(i) at least one element of $A$ divides another, and

(ii) at least two elements of $A$ are coprime.

**Proof**

Both use the pigeonhole principle, although for (ii) it is a slightly modified version.

(i) Pick out the largest powers of 2 which divide each element of $A$; that is, write the elements of $A$ as $2^{a_1}b_1, 2^{a_2}b_2, \ldots, 2^{a_n}b_n$, where all the $b_i$ are odd and all the $a_i \in \mathbb{N}$. All the $b_i$ are odd numbers less than $2n$: there are only $n$ such odd numbers, and $n+1$ elements of $A$, so we must have $b_i = b_j$ for some $i, j$. Of the corresponding elements of $A$, one is a power of 2 times the other.

(ii) We need a modified pigeonhole principle: you cannot put more than $n$ pigeons into $2n$ *different* pigeonholes without having at least two pigeons in consecutive holes.

Here, we have more than $n$ distinct positive integers fitting into $2n$ slots, so two must be consecutive, and these must be coprime because any common divisor of $a$ and $b$ also divides their difference. ■

Finally, a pure number theory result from the pigeonhole principle.

**Claim 6.7** For any prime $p$ there is a solution (in $x$ and $y$) of $x^2+y^2 \equiv -1 \pmod{p}$.

**Proof** Equivalently, we must prove that there is a solution of $x^2+1 \equiv -y^2 \pmod{p}$.

$x_1^2 \equiv x_2^2 \pmod{p}$ if and only if $p \mid (x_1 - x_2)(x_1 + x_2)$, if and only if $x_1 \equiv \pm x_2 \pmod{p}$. So there are exactly $\frac{p+1}{2}$ different values (mod $p$) for $x^2+1$ and exactly $\frac{p+1}{2}$ different values for $-y^2$. There are only $p$ different values (mod $p$), and $p+1$ values of $x^2+1$ or $-y^2$ so the pigeonhole principle tells us that at least one pair must be equal. We already said that the $\frac{p+1}{2}$ values for each of $x^2+1$ were different, similarly for $-y^2$, so one of the $x^2+1$ must equal one of the $-y^2$. This proves that $x^2+1 \equiv -y^2 \pmod{p}$ for some $x$ and $y$. ■

## 6.6 Interesting Diversion: Modular Square Roots of -1

For a bonus, we prove a pure number theoretic result using the facts connecting equivalence relations and partitions.

**Claim 6.8** If $p$ is prime, then $x^2 \equiv -1 \pmod{p}$ has a solution if and only if $p = 2$ or $p \equiv 1 \pmod{4}$.

**Proof** If $p = 2$ then $x = 1$ is a solution (in fact the only solution, up to equality (mod 2)).

For odd $p$, we construct an equivalence relation on $\{1, 2, \ldots, p-1\}$:

$$y_1 \sim y_2 \quad \text{if} \quad y_1 \equiv \pm y_2 \pmod{p} \text{ or } y_1 \equiv \pm y_2^{-1} \pmod{p}.$$

It is easy to check that this does indeed define an equivalence relation. Each equivalence class of $\sim$ can have up to four elements: $\{y, -y, y^{-1}, -y^{-1}\}$, but it is possible that these are not four distinct numbers:

(i) It is impossible to have $y \equiv -y \pmod{p}$ when $p$ is odd.

(ii) $y \equiv y^{-1} \pmod{p}$ and $-y \equiv -y^{-1} \pmod{p}$ (one of these happens if and only if the other does). These are equivalent to $y^2 \equiv 1 \pmod{p}$, i.e. $p \mid (y^2 - 1) = (y-1)(y+1)$. This happens only when $y \equiv \pm 1 \pmod{p}$, so there is always one equivalence class with exactly two elements: $\{1, -1\}$.

(iii) $y \equiv -y^{-1} \pmod{p}$ and $-y \equiv y^{-1} \pmod{p}$ (again, each of these implies the other). They are equivalent to $y^2 \equiv -1 \pmod{p}$. There might be a solution to this equation or there might not, but there can only be zero or two solutions to equations of this form (proof: $y^2 \equiv a$ is equivalent to $(y-b)(y+b) \equiv 0$: $y \equiv \pm b$ are the only solutions, if any such $b$ exists at all), so there may be one further equivalence class of size two $\{y, -y\}$, or there may be no further equivalence class of size two.

So there are always one (if $y^2 \equiv -1$ has no solution) or two (if $y^2 \equiv -1$ has two solutions) equivalence classes of size 2, and all the other equivalence classes must be of size 4.

Now remember that the equivalence classes partition the set on which the equivalence relation is defined, here $\{1, 2, \ldots, p-1\}$. The cardinality is $p-1$,

and so the cardinality of the equivalence classes adds up to $p-1$. If $p-1 \equiv 0 \pmod 4$ then there must be two equivalence classes of size 2, so there is a $y$ satisfying $y^2 \equiv -1 \pmod p$. On the other hand, if $p - 1 \equiv 2 \pmod 4$ then there must be one equivalence class of size 2, so there is no solution of $y^2 \equiv -1 \pmod p$. ■

## Practice Questions

**6.1** Compute the remainder when $(34 \cdot 45 \cdot 57 + 89)^2$ is divided by 8.

**6.2** Compute the remainder when $13^{17}$ is divided by 11.

**6.3** Use Euclid's Algorithm to find the gcd of 805 and 105.

**6.4** Use Euclid's Algorithm to find integers $x$ and $y$ satisfying $21x + 99y = 3$.

**6.5** Show that $\gcd(F_{n+1}, F_n) = 1$ (where $F_n$ is the $n$-th Fibonacci number). Use proof by induction on $n$.

**6.6** Show that 14 cannot be written as the sum of two squares. Write it as the sum of three squares.

**6.7** Consider an equilateral triangle which has sides of length 1 unit, and five points placed in the triangle. Show that, no matter where the points are placed, at least two of them must be no more than $\frac{1}{2}$ units apart.

**6.8** Suppose we are given a finite set $S \subset \mathbb{N}$. If $\log_2 n < |S|$, show that there exist two distinct subsets of $S$, $S_1$ and $S_2$, satisfying

$$\prod_{a \in S_1} a = \prod_{a \in S_2} a \pmod{n}.$$

(Recall that $\prod_{a \in S_1} a$ means the product of all members of $S_1$.)

## Answers to Chapter 6 Practice Questions

**6.1** According to the laws for addition, multiplication, and exponentiation (mod 8), we have $34 \equiv 2 \pmod 8$, $45 \equiv 5 \pmod 8$, $55 \equiv -1 \pmod 8$, $89 \equiv 1 \pmod 8$, so $(34 \cdot 45 \cdot 55 + 89)^2 \equiv (2 \cdot 5 \cdot -1 + 1)^2 \equiv (-9)^2 \equiv (-1)^2 \equiv 1 \pmod 8$.

**6.2** Again, we have $13^{17} \equiv 2^{17} \pmod{11}$. But we *cannot* reduce the exponent 17 to 6 modulo 11. Use the method of repeated squaring: $2^{17} \equiv 2 \cdot 2^{16} \equiv 2 \cdot 4^8 \equiv 2 \cdot 16^4 \equiv 2 \cdot 5^4 \equiv 2 \cdot 25^2 \equiv 2 \cdot 3^2 \equiv 18 \equiv 7 \pmod{11}$.

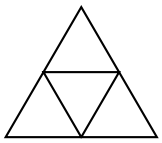
**6.3** 35.

**6.4** $21 \cdot (-14) + 99 \cdot 3 = 3$.

**6.5** The base case is easy: $\gcd(F_2, F_1) = \gcd(1, 1) = 1$. For the inductive step, use the fact that $\gcd(m, n) = \gcd(m, n - m)$ (which is part of Euclid's Algorithm) and commutativity of $\gcd(-, -)$. We have $\gcd(F_{n+1}, F_n) = \gcd(F_n + F_{n-1}, F_n) = \gcd(F_n, F_{n-1})$. By the inductive hypothesis, this equals 1.

**6.6** The only squares less than 14 are 1, 4, and 9. No pair from this set sums to 14, but all three do. So $14 = 1^2 + 2^2 + 3^2$.

**6.7** We need to turn this into a pigeonhole principle situation. Let us divide up the triangle into four regions, equilateral of size $\frac{1}{2}$ units:



With four regions and five points, at least two points must lie in the same region. Those two points cannot be further apart than the vertices of the small triangle, a maximum of $\frac{1}{2}$ units.

**6.8** If $\log_2 n < |S|$ then $n < |\mathcal{P}(S)|$. There are only $n$ different values for *any* function (mod $n$), so two different subsets of $S$ must give rise to the same product (mod $n$).

# Chapter 7

# Asymptotic Notation

---

**Reading:** Ross & Wright: 4.3;
Chetwynd & Diggle: not covered;
Grimaldi: 5.7, 5.8, 10.6;
Grossman: 13.4 (mostly not covered).

---

The asymptotic behaviour of a function indicates the rate at which it "grows", for "large" inputs. In many computer science applications, particularly when counting the time taken or memory used by an algorithm, we are happy to know the rate of growth of a function without knowing the function exactly. This is partly because our measures of time are usually approximate anyway, counting computational steps rather than something necessarily directly related to seconds. Moreover, quite often we cannot compute exact solutions for the recurrence relations which give the time or space requirements of an algorithm, but we can at least determine the rate of growth. This chapter discusses one of the notations used for rates of growth, and explores its properties as an interesting example of a particular type of mathematical statement which needs a carefully-constructed proof.

## 7.1 Big-O Notation

When we say that one function grows asymptotically no faster than another, we do not mean that it must always be smaller. Asymptotic growth is

measured only: a) without regard to constant multiples, and b) only for sufficiently large values of the domain. Formally,

**Definition** If $f, g : \mathbb{N} \to \mathbb{R}$, we write $f(n) = O(g(n))$ if there are constants $c \in \mathbb{R}$ and $N \in \mathbb{N}$ satisfying

$$|f(n)| \leq c|g(n)| \text{ for all } n \geq N. \tag{7.1}$$

Recall that $|x|$ denotes the **absolute value** of $x$, $|x| = x$ if $x \geq 0$ and $|x| = -x$ if $x < 0$. In computer science the definition is most commonly used when $f$ and $g$ are guaranteed to give positive values (because they represent time or memory), in which case the modulus signs $|-|$ can be dropped. The same definition works for sequences – read $f_n$ instead of $f(n)$ – and also functions with domains such as $\mathbb{R}$ or $[0, \infty)$ (in which case $N$ is a member of the same set). Sometimes, when the functions involved already have names, we elide the function argument $n$, writing $f = O(g)$ instead.

This is called **big-O notation** or **Landau's notation** (after the mathematician who popularized its use). We also say that $f$ is **asymptotically bounded** or **asymptotically dominated** by $g$.

For an example, $n^2 + n = O(n^2)$. This is true because $n^2 + n \leq 2n^2$ for $n \geq 1$, which is true because $n^2 \geq n$ for $n \geq 1$. Everything is positive so we can forget about the $|-|$ signs.

We also have $100n^2 + 10000n = O(n^2)$. In this case we need a larger constant $c$ (10100 will do). You can see that multiplying a function by a constant, no matter how large, does not affect its "big-O" behaviour.

For another example, $n^3 + n^2 + \log n = O(n^3)$. This is true because $n^3 \geq n^2$ for $n \geq 1$ and $n^3 \geq \log n$ for all $n > 0$ (how could you prove the latter?). Combining these facts, $n^3 + n^2 + \log n \leq 3n^3$ for $n \geq 1$ (again, everything is positive so the $|-|$ signs are irrelevant).

We have $n^5 = O(n^n)$, because $n^5 \leq 1 \cdot n^n$ for $n \geq 5$. It does not matter that the inequality fails for smaller values of $n$; only the behaviour for large $n$ matters. We also have $n! = O(n^n)$: this is true because $n! = 1 \cdot 2 \cdots (n-1) \cdot n$, a product of $n$ terms each $\leq n$. Therefore it is less than $n^n$. We have proved $n! < 1 \cdot n^n$ for all $n \geq 1$. But this is not a *tight* upper bound: in the same way that $n^2 \leq n^4$ is true but not tight as, and less informative than, $n^2 \leq n^2$. In Sect. 7.4 we will find a tighter bound for the asymptotic behaviour of $n!$.

For a final example, $3^{\log n} = O(n^2)$. This is true because

$$3^{\log n} = e^{(\log n)(\log 3)} = n^{\log 3}.$$

Now $\log 3 < 2$, so $n^{\log 3} \leq n^2$ for all $n \geq 1$, completing the proof.

Now one tricky aspect of big-O notation is that $f(n) = O(g(n))$ is not an equality, even though it uses the equals sign. In fact it works more like "$\leq$". If we know that $f_1(n) = O(g(n))$ and $f_2(n) = O(g(n))$, we *cannot* conclude that $f_1 = f_2$: the preceding examples demonstrate this. Neither can we switch between the two sides: $n = O(n^2)$ but $n^2 \neq O(n)$. But if $f = O(g(n))$ and $g = O(h(n))$ then $f = O(h(n))$ (this is an exercise). Some books write $f(n) \in O(g(n))$, so that $O(g(n))$ represents the set of all functions $f$ which satisfy the equation (7.1). That notation avoids confusion with equality, but we will not use it here.

To decide whether given $f$ and $g$ satisfy $f(n) = O(g(n))$, it is often useful to examine the quotient $f(n)/g(n)$. If this quantity can be **bounded** (its absolute value is always less than some value $c$), for sufficiently large $n$, then $f(n) = O(g(n))$ is established. If it cannot be bounded then $f(n) \neq O(g(n))$. It is also helpful to establish some algebraic laws for big-O notation, and some of these are explored in the practice questions and tutorial exercises.

One other technique, sometimes useful for showing $f(n) = O(g(n))$, is to take logs of the quotient: assuming that everything in sight is positive, $f(n)/g(n)$ is bounded if and only if $\log(f(n)/g(n)) < k$ for some $k$. This can simplify big-O problems involving exponentials. We will see an example of this in the next section.

## 7.2 Proving Sentences of the form $\exists x.\forall y.P$

The statement $f(n) = O(g(n))$ is of a form commonly found in mathematics and computer science. Using logical notation, we can write it $\exists x.\forall y.P$. In this case, **there exists** a pair of numbers $c$ and $N$, such that **for all** $n$, whenever $n \geq N$ then $|f(n)| \leq c|g(n)|$. How do we prove such a statement?

The easiest form of proof is to work out (somehow) what the numbers $c$ and $N$ must be, and then prove that $n \geq N$ implies $|f(n)| \leq c|g(n)|$ using the usual techniques. For example, to prove that $(n+1)^3 = O(n^3)$ we might see

straightaway that $c = 8$ and $N = 1$ will work, then construct the following proof:

Suppose $n \geq 1$. Then $2n \geq n + 1$, so $8|n^3| \geq |(n + 1)^3|$. End of proof.

But, in practice, we often cannot find the right values of $c$ and $N$ immediately. In that case, it is better to work some calculations into the proof, beginning with as-yet undetermined values of $c$ and $N$ and hoping to derive them as the proof progresses. In that case, take care that the logic is correct: the proof is being constructed backwards, and probably the steps in the proof are $\Leftarrow$ or $\Leftrightarrow$ (not $\Rightarrow$). For example,

**Claim 7.1** $n^2 = O(2^n)$.

**Proof** Everything in sight is positive, so the claim is true if (and only if), for some $N$ and $c$, whenever $n \geq N$ we have $n^2 \leq c2^n$. Then calculate:

$$\begin{aligned} n^2 \leq c2^n &\Leftrightarrow \tfrac{n^2}{2^n} \leq c \\ &\Leftrightarrow 2\log n - n\log 2 \leq \log c \end{aligned}$$

We can see that we need to find a bound on $2\log n - n\log 2$. There are a number of ways, but we will use a simple method involving some continuous mathematics.

Consider the function $f : (0, \infty) \to \mathbb{R}$, $f(x) = 2\log x - x\log 2$. We compute $f'(x) = 2/x - \log 2$ and $f''(x) = -2/x^2$. Therefore $f$ has one turning point, a maximum, at $x = 2/\log 2$. It is enough to notice that $2/\log 2 < 4$, and $f(4) = 2\log 4 - 4\log 2 = 0$, so at the very least we have $f(x) < 0$ for $x \geq 4$.

Now we have found $N$ and $c$: if we take $N = 4$ and $c = 1$, we have just shown that $2\log n - n\log 2 \leq \log c$ for $n \geq N$. Following the calculation backwards completes the proof. ■

The same techniques apply to proofs of other sentences of the form $\exists x.\forall y.P$: although one can sometimes write down $x$ straight away, it is often necessary to begin the proof with $x$ undetermined, and later deduce the value which will make the proof work.

(We might sometimes need to prove $f(n) \neq O(g(n))$, and that is a mathematical statement of a different shape. These will be examined in the final chapter, but for now we need only note that it is sufficient to show that $|f(n)/g(n)|$ is *not* bounded.)

## 7.3 Tail Behaviour

Now we prove a result about big-O notation. It is an example of how to construct proofs involving sentences of the form $\exists x.\forall y.P$, and also shows us a slightly simpler equivalent definition for $f = O(g)$.

**Claim 7.2** As long as the domain and $f$ and $g$ is $\mathbb{N}$, and $g(n)$ is always nonzero, $f(n) = O(g(n))$ if and only if there is a constant $c \in \mathbb{R}$ satisfying

$$|f(n)| \leq c|g(n)| \text{ for all } n \in \text{Dom}(f).$$

**Proof** The direction ($\Leftarrow$) is trivial: if there is a $c$ satisfying $|f(n)| \leq c|g(n)|$ for all $n$, then just take $N = 0$ (or any $N \in \text{Dom}(f)$); we already have the same inequality for $n \geq N$.

For ($\Rightarrow$), suppose that there exist $c$ and $N$ such that $|f(n)| \leq c|g(n)|$ for all $n \geq N$. We will need to find a $c'$ such that $|f(n)| \leq c'|g(n)|$ for all $n$, regardless of whether $n \geq N$ or $n < N$. We can write down such a number:

$$c' = \max\{|f(0)/g(0)|, |f(1)/g(1)|, |f(2)/g(2)|, \ldots, |f(N-1)/g(N-1)|, c\}$$

Let us prove that $c'$ has the required property. Because $c' \geq |f(n)/g(n)|$ for $n < N$, we have $|f(n)| \leq c'|g(n)|$ for $n < N$. And because $c' \geq c$, we have $|f(n)| \leq c'|g(n)|$ for $n \geq N$ too. That completes the proof. ■

A similar result extends to functions $f$ and $g$ whose domain is $[0, \infty)$, but the proof is more difficult. We must ensure that $|f(n)/g(n)|$ has a maximum on $[0, N)$. This is guaranteed as long as $f$ and $g$ are **continuous** functions, but that takes us well away from discrete mathematics.

## 7.4 Example: Asymptotics of $n!$

Now let us look at an advanced example. We have already commented that $n! = O(n^n)$, but this is not a very good bound: only one of the products in $n!$ is as big as $n$, so it seems that $n^n$ significantly overestimates $n!$. To give a more precise description of the asymptotic behaviour of $n!$, we must delve into continuous mathematics.

First, a simple lemma. (A **lemma** is a mathematical statement which is proved as a preliminary step towards the proof of a bigger claim.)

**Lemma 7.3** For all $x > 0$, $(\frac{1}{2} + \frac{1}{x}) \log(1 + x) > 1$.

**Proof** The result is equivalent to $\log(1+x) > \frac{x}{\frac{1}{2}x+1}$. Consider the function $f : [0, \infty) \to \mathbb{R}$, $f(x) = \log(1 + x) - \frac{x}{\frac{1}{2}x+1}$. Differentiating, we have

$$f'(x) = \frac{1}{1+x} - \frac{\frac{1}{2}x + 1 - \frac{1}{2}x}{\left(\frac{1}{2}x + 1\right)^2} = \frac{1}{1+x} - \frac{1}{1 + x + \frac{1}{4}x^2} > 0$$

for $x > 0$, the last inequality because $1 + x + \frac{1}{4}x^2 > 1 + x$. Therefore $f$ is strictly increasing, and we know that $f(0) = 0$, so $f(x) > 0$ for all $x > 0$. This is precisely the result we want. ■

Now we can state and prove a tight asymptotic bound for the factorial function.

**Claim 7.4**

$$n! = O(n^{n+\frac{1}{2}} \exp(-n)).$$

**Proof** First, define $a_n = \frac{n!}{n^{n+\frac{1}{2}} \exp(-n)}$; we need to find a constant $c$ such that $a_n \leq c$, at least for sufficiently large $n$.

Now consider $\frac{a_n}{a_{n+1}}$. We have

$$\begin{aligned} \frac{a_n}{a_{n+1}} &= \frac{n!(n+1)^{n+1+\frac{1}{2}} \exp(-n-1)}{(n+1)!n^{n+\frac{1}{2}} \exp(-n)} \\ &= \frac{1}{e}\Big(1 + \frac{1}{n}\Big)^{n+\frac{1}{2}} \end{aligned}$$

so $\log\big(\frac{a_n}{a_{n+1}}\big) = (\frac{1}{2} + n) \log(1 + \frac{1}{n}) - 1$. According to the previous lemma, this is positive at least for $n \geq 1$. Therefore

$$\frac{a_n}{a_{n+1}} > 1$$

or $a_{n+1} < a_n$ for $n \geq 1$. We have shown that $(a_n)$ is a decreasing sequence, so $a_{n+1} < a_1$, so $c = a_1$ (which happens to equal $e$) satisfies $a_n \leq c$ for all $n \geq 1$. ■

It turns out that this asymptotic bound is as good as possible: in fact, $\frac{n!}{n^{n+\frac{1}{2}} \exp(-n)}$ tends to a finite value $(\sqrt{2\pi})$ as $n \to \infty$. That is known as

**Stirling's formula**, but its proof requires more continuous mathematics than we want to see in this course. Stirling's formula gives the following approximation to $n!$, for large $n$: $n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$.

This sort of proof, manipulating properties of continuous functions to obtain a bound, is typical of the methods used to obtain asymptotic behaviour of complicated functions or sequences.

## 7.5 Example: Asymptotic Behaviour of Recurrence Relations

We already know that recurrence relations, especially those which are non-linear, can be difficult to solve. But often, such as for recurrences describing the running time of an algorithm, we are content to bound the asymptotic growth of the solution rather than finding the solution itself. This is sometimes an easier task.

For example, consider the recurrence

$$x_1 = 0, \quad x_n = 2x_{\lfloor \frac{n}{2} \rfloor} + n \text{ for } n \geq 2$$

The sequence it generates begins $(0, 2, 3, 8, 9, 12, 13, 24, 25, 28, \ldots)$ and we might well have difficulty finding a closed formula for the $n$-th term. But we can prove

$$x_n = O(n \log_2 n)$$

using induction, although the induction is a bit strange because when we start it we do not know exactly what we are going to prove[1].

The base case, at least, will be automatic: $x_1 = 0$ is less than any multiple of $n \log_2 n$.

Now consider the inductive step. We want to prove that $x_n \leq cn \log_2 n$, for some value $c$, but we don't yet know the value of $c$. Nonetheless, we will attempt a proof by induction with $c$ as an arbitrary value, and aim to fill it in later.

[1]Because big-O notation is insensitive to constant multiples, and because $\log_b n = (\log_b a)(\log_a n)$, we do not actually need to specify the base of the logarithms in situations like this.

Using the recurrence, if $n \geq 2$ we have

$$\begin{aligned} x_n &= 2x_{\lfloor \frac{n}{2} \rfloor} + n \\ &\leq 2c\lfloor \tfrac{n}{2} \rfloor \log_2 \lfloor \tfrac{n}{2} \rfloor + n \quad \text{(IH)} \\ &\leq cn \log_2(n/2) + n \\ &= cn(\log_2 n - 1) + n \\ &= cn \log_2 n + n(1-c) \end{aligned}$$

now as long as $c \geq 1$, this final formula is less than $cn \log_2 n$, which completes the inductive step. So it the proof by induction, of the statement $x_n \leq cn \log_2 n$, works for any $c \geq 1$.

We should comment that things are more difficult if the initial condition is modified to $x_1 = 1$. Then it is still true that $x_n = O(n \log_2 n)$. But the base case cannot possibly be $n = 1$, because $c \cdot 1 \log_2 1 = 0$, not a number at least 1. Of course, for asymptotic behaviour we only need prove that $x_n \leq n \log_2 n$ for sufficiently large $n$, so we can move the base case to $n = 2$. This requires us to use a larger value of $c$, but (looking carefully at the structure of the induction) also another base case $n = 3$. If you can understand why, you will have grasped the finer details of proof by induction.

## 7.6 Interesting Diversion: Solving Recurrences of the Divide-and-Conquer Type

Recurrences of the form

$$t_n = at_{\lfloor \frac{n}{b} \rfloor} + f(n),$$

for constants $a \geq 1$ and $b \geq 1$, and a function $f(n)$, occur often in the analysis of algorithms (when a problem of size $n$ is split into $a$ smaller problems, each size $n/b$), and they can be difficult to solve exactly. However the asymptotic behaviour of the solution can often be deduced immediately (and usually regardless of the initial conditions, in fact). There are standard answers in many cases, and we will present them here without proof.

First, we need to extend big-O notation. Recall that $f = O(g)$ is somewhat akin to "$f \leq g$". The converse, akin to "$f \geq g$", and something similar to equality, are also defined:

**Definition** We write

$$f(n) = \Omega(g(n))$$

if $g(n) = O(f(n))$, and

$$f(n) = \Theta(g(n))$$

if both $f(n) = O(g(n))$ and $g(n) = O(f(n))$.

We will not study so-called big-Omega and big-Theta notation in this course, but note that $f(n) = \Theta(g(n))$ gives the most precise description of the asymptotic behaviour of $f(n)$: it is bounded above and below by constant multiples of $g(n)$, for sufficiently large $n$.

We now state a theorem which allows us to write down solutions to recurrences of the type we described. Along with a proof, it can be found in the standard algorithms textbook

T. H. Cormen, C. E. Leiserson, R. L Rivest. *Introduction to Algorithms*, MIT Press 1990.

where it is called the "master theorem".

**Theorem 7.5** Consider the recurrence

$$t_n = at_{\frac{n}{b}} + f(n)$$

where $t_{\frac{n}{b}}$ can mean either $t_{\lfloor \frac{n}{b} \rfloor}$ or $t_{\lceil \frac{n}{b} \rceil}$, and the boundary conditions can be anything as long as they ensure that $t_n$ is positive (at least for sufficiently large $n$). Then

(i) If $f(n) = O(n^k)$ with $k < \log_b a$, then $t_n = \Theta(n^{\log_b a})$.

(ii) If $f(n) = \Theta(n^k)$ with $k = \log_b a$, then $t_n = \Theta(n^{\log_b a} \log n)$.

(iii) If $f(n) = \Omega(n^k)$ with $k > \log_b a$, then $t_n = \Theta(f(n))$.

The theorem tells us whether the recurrence part $at_{\frac{n}{b}}$ is most important to the behaviour of the sequence, or the part $f(n)$, or whether both contribute (the middle case). For example, we can now say that the asymptotic behaviour of the three recurrence relations

$$s_n = 3s_{\lfloor \frac{n}{2} \rfloor} + n, \quad t_n = 4t_{\lfloor \frac{n}{2} \rfloor} + n^2, \quad u_n = 5u_{\lfloor \frac{n}{2} \rfloor} + n^3,$$

(whatever their initial conditions, as long as they are enough to make the resulting sequences positive) are

$$s_n = \Theta(n^{\log_2 3}), \quad t_n = \Theta(n^2 \log n), \quad u_n = \Theta(n^3).$$

---

## Practice Questions

**7.1** Which of the following statements are true? There is no need to include proofs.

(i) $n^2 = O(n^3)$,
(ii) $10n + 5 = O(n)$,
(iii) $n^2 + 1 = O(n)$,
(iv) $\sqrt{n^4 + n^2} = O(n^2)$,
(v) $100000 + \log n = O(1)$.

**7.2** Give a sketch proof that:

(i) $\binom{n}{2} = O(n^2)$ (the domain here must be $\{n \in \mathbb{N} \mid n \geq 2\}$),
(ii) $\binom{n}{3} = O(n^3)$ (the domain here must be $\{n \in \mathbb{N} \mid n \geq 3\}$).

**7.3** Which of the following statements are true? Explain why.

(i) $n2^n = O(2^n)$,
(ii) $n2^n = O(3^n)$,
(iii) $n^3 = O(3^n)$,
(iv) $3^n = O(n^3)$.

**7.4** Show that, for any $a \in \mathbb{R}$ and $b \in [0, \infty)$, $(n + a)^b = O(n^b)$.

What is the best power of $n$ asymptotic bound for $(n + a)^b$ if $a \neq 0$ and $b < 0$?

**7.5** Define two functions $f, g : \mathbb{N}_+ \to \mathbb{N}_+$ by

$$f(n) = \begin{cases} n, & \text{for } n \text{ odd} \\ 1, & \text{for } n \text{ even} \end{cases} \qquad g(n) = \begin{cases} 1, & \text{for } n \text{ odd} \\ n, & \text{for } n \text{ even} \end{cases}$$

Prove that $f \neq O(g)$ and $g \neq O(f)$.

**7.6** Prove that $f = O(g)$ and $g = O(h)$ together imply $f = O(h)$.

**7.7** Prove that, if $f_1 = O(g_1)$ and $f_2 = O(g_2)$, *and the functions* $f_1$, $f_2$, $g_1$, $g_2$ *are nonnegative*, then $f_1 + f_2 = O(g_1 + g_2)$.

**7.8** Consider the recurrence $x_n = 2x_{\lfloor \frac{n}{2} \rfloor} + n^2$, $x_1 = 1$. Without using the theorem in Sect. 7.6, prove that $x_n = O(n^2)$.

## Answers to Chapter 7 Practice Questions

**7.1** (i), (ii), and (iv) are true.

**7.2** $\binom{n}{2} = \frac{n(n-1)}{2} \le n^2$ and $\binom{n}{3} = \frac{n(n-1)(n-2)}{6} \le n^3$.

**7.3** (i) false: the quotient $n2^n/2^n = n$ is not bounded.
(ii) true: take the log of the quotient: $\log(n2^n/3^n) = \log n - n\log(3/2)$, differentiate to find that it has a maximum.
(iii) true: take the log of the quotient: $\log(n^3/3^n) = 3\log n - n\log 3$, differentiate to find that it has a maximum.
(iv) false: take the log of the quotient: $\log(3^n/n^3) = n\log 3 - 3\log n$, its derivative $\log 3 - 3/n$ is positive and increasing for $n \ge 1$, so the quotient must grow without bound.

**7.4** The key is: for $n \ge a$, $(n+a)^b \le (n+n)^b = 2^b n^b$. If $b < 0$ and $a \ne 0$, the best we can say is $(n+a)^b = O(1)$.

**7.5** $f(n)/g(n) = \begin{cases} n, & \text{for } n \text{ odd} \\ 1/n, & \text{for } n \text{ even} \end{cases}$ and $g(n)/f(n) = \begin{cases} 1/n, & \text{for } n \text{ odd} \\ n, & \text{for } n \text{ even} \end{cases}$.
No matter what value we pick for $c$ or $N$, there is always an odd integer $n$ bigger than both so $f(n)/g(n)$ is not bounded by any multiple of $n$; similarly for $g(n)/f(n)$.

**7.6** By the hypotheses, there exist $c_1, c_2, N_1, N_2$ such that $|f(n)/g(n)| \le c_1$ for $n \ge N_1$ and $|g(n)/h(n)| \le c_2$ for $n \ge N_2$. Set $N = \max\{N_1, N_2\}$: then for $n \ge N$ we have both inequalities, and multiplying them gives $|f(n)/h(n)| \le c_1 c_2$. So, taking $c = c_1 c_2$, we have shown that $f = O(h)$.

**7.7** Since everything is positive, we can drop the $|-|$ signs. Because $f_1 = O(g_1)$, there exist $c_1$ and $N_1$ such that $f_1(n)/g_1(n) \le c_1$ for $n \ge N_1$. And because $f_2 = O(g_2)$, there exist $c_2$ and $N_2$ such that $f_2(n)/g_2(n) \le c_2$ for $n \ge N_2$. Then

$$\frac{f_1(n)+f_2(n)}{g_1(n)+g_2(n)} = \frac{f_1(n)}{g_1(n)+g_2(n)} + \frac{f_2(n)}{g_1(n)+g_2(n)} \le \frac{f_1(n)}{g_1(n)} + \frac{f_2(n)}{g_2(n)} \le c_1 + c_2$$

as long as $n \ge \max\{N_1, N_2\}$. Therefore $c = c_1 + c_2$ and $N = \max\{N_1, N_2\}$ prove $f_1 + f_2 = O(g_1 + g_2)$.

**7.8** Try to prove, by induction, that $x_n \le cn^2$ (for some $c$ yet to be determined). For the inductive step, we have $x_n \le 2c\left\lfloor \frac{n}{2} \right\rfloor^2 + n^2 \le n^2(\frac{c}{2}+1) \le cn^2$ as long as $c \ge \frac{c}{2} + 1$. $c = 2$ will do. The base case is easy to check.

# Chapter 8

# Orders

---

**Reading:** Ross & Wright: 11.1, 11.2;
Chetwynd & Diggle: 3.3 (barely covered);
Grimaldi: 7.3;
Grossman: 5.5 (mostly not covered).

---

Orders are particular types of relation with a simple intuitive interpretation, formalizing the idea of elements being "greater than" or "after" others. But orders need not be as simple as $\leq$ on $\mathbb{R}$, and even with this familiar order there are some quite subtle concepts related to the idea of maximum: these will illustrate our final proof topic.

Although they might seem a little dry at first sight, orders are of great importance to computer scientists: orders, and some related concepts, are pervasive in theoretical computer science, so it is worthwhile to meet them here.

## 8.1 Definitions

There are three commonly-used types of order, all of them relations with certain properties.

**Definition** A **preorder** is a reflexive, transitive relation.

A **partial order** is a reflexive, antisymmetric, transitive relation.

A **linear order** is an antisymmetric, transitive relation with the additional property of **totality**: for every $x, y$ (in the domain) either $x\,R\,y$ or $y\,R\,x$ (or both).

A set together with a partial order on that set is sometimes called a **partially ordered set** (also called **poset**), similarly for a **totally ordered set**. Some books use the terminology **total order** instead of linear order, and **chain** instead of totally ordered set. But, as we shall see shortly, the word chain also has other (similar) meanings. We tend to use the generic term **order** to mean a set that has a preorder, partial order, or linear order.

Note that the totality condition for linear orders also implies reflexivity, so the linear orders are a subset of the partial orders, which are a subset of the preorders.

Some familiar examples: on any subset of $\mathbb{R}$, $\leq$ is a linear order (hence a partial order and a preorder); on $\mathbb{N}_+$, $\mid$ is a partial order and a preorder but not a linear order (neither $2 \mid 3$ nor $3 \mid 2$); for any fixed set $A$ there is an order on $\mathcal{P}(A)$ given by $\subseteq$ (again a partial order but, if $A$ has at least two elements, not a linear order); there is an alternative order on $\mathcal{P}(A)$ given by $B \leq C$ if $|B| \leq |C|$ but this is only a preorder since it is not antisymmetric. We could order sequences of letters by "dictionary order" (the technical term is lexicographic order, of which more later) to make a linear order on English words. Another order on English words is given by **prefix**: $w_1 \preceq w_2$ if $w_1$ is an initial segment of $w_2$ (e.g. "line" $\preceq$ "linear"); this is a partial order but not a linear order.

It is easy to see that the converse of a preorder (partial order, linear order) is still a preorder (partial order, linear order), so all of the preceding orders can also be reversed. We have to be careful with orders which are not linear: if $x \neq y$, and $\succeq$ the converse of $\preceq$, it is tempting to confuse $x \not\preceq y$ with $x \succeq y$. Unless $\preceq$ is a linear order, they are not always the same: we have $2 \nmid 3$ but we cannot conclude $3 \mid 2$.

**Definition** Suppose that $\preceq$ is an order on a set $A$. If either $x \preceq y$ or $y \preceq x$ then we say that $x$ and $y$ are **comparable**, otherwise they are **incomparable**.

When $\preceq$ is a partial order on $A$, we say that $S \subseteq A$ is a **chain** if all pairs in $S$ are comparable, and $S \subseteq A$ is an **antichain** if no pairs in $S$ are comparable.

(To further confuse matters, sometimes the word **chain** means a finite or infinite *sequence* $x_1, x_2, \ldots$ of elements of $A$ with $x_1 \preceq x_2 \preceq \cdots$, but we will not use that terminology here.)

The totality condition of a linear order means that all elements are comparable, making these definitions uninteresting. But we can find some interesting examples using the partial order $|$ on $\mathbb{N}$: with respect to this order, both $\{1, 3, 9, 27, 81\}$ and $\{1, 2, 4, 12, 84, 168\}$ are examples of chains and both $\{2, 3, 5, 7\}$ and $\{7, 8, 9, 10, 11, 12, 13\}$ are examples of antichains.

## 8.2 Orders on Cartesian Products

Suppose that we have an order on $A$; can we construct an order on $A \times A$? There are a number of alternatives. First, some useful terminology. We will write $x \prec y$ to mean that $x \preceq y$ and $x \neq y$: this irreflexive so-called **strict order** can be created from any preorder, partial order, or linear order.

The two commonest product orders are

**Definition** If $\preceq$ is an order on $A$ then the **lexicographic order** on $A \times A$ is defined by

$$(x, y) \preceq_L (x', y') \quad \Leftrightarrow \quad x \prec x' \text{ or } (x = x' \text{ and } y \preceq y').$$

If $\preceq$ is an order on $A$ then the **product order** on $A \times A$ is defined by

$$(x, y) \preceq_P (x', y') \quad \Leftrightarrow \quad x \preceq x' \text{ and } y \preceq y'.$$

The product order is more strict, in the sense that $(x, y) \preceq_P (x', y')$ implies $(x, y) \preceq_L (x', y')$ but not vice versa. The lexicographic order is the same as "dictionary order" on words of fixed length. The properties of the lexicographic construction match those of the order to which it is applied, but this is not always true of the product order.

**Claim 8.1**

(i) If $\preceq$ is a preorder on $A$ then $\preceq_L$ is a preorder on $A \times A$.

(ii) If $\preceq$ is a partial order on $A$ then $\preceq_L$ is a partial order on $A \times A$.

(iii) If $\preceq$ is a linear order on $A$ then $\preceq_L$ is a linear order on $A \times A$.

(iv) If $\preceq$ is a preorder on $A$ then $\preceq_P$ is a preorder on $A \times A$.

(v) If $\preceq$ is a partial order on $A$ then $\preceq_P$ is a partial order on $A \times A$.

(vi) If $\preceq$ is a linear order on $A$ then $\preceq_P$ **might not be** a linear order on $A \times A$.

**Proof** We need to show that whenever $\preceq$ is reflexive (respectively transitive, antisymmetric) the same is true for $\preceq_L$ and $\preceq_P$. Also, if $\preceq$ is total then so is $\preceq_L$.

Let us show that both constructions preserve reflexivity. Suppose that $\preceq$ is reflexive on $A$, and consider any pair $(x, y) \in A^2$. Since $x \preceq x$ and $y \preceq y$, we have $(x, y) \preceq_P (x, y)$. For the lexicographic order we have a similar argument: at the final stage, we know that $x = x$ and $y \preceq y$, thus $(x, y) \preceq_L (x, y)$.

Now consider antisymmetry. Suppose that $(x, y) \preceq_P (x', y') \preceq_P (x, y)$. Then we must have $x \preceq x' \preceq x$ and $y \preceq y' \preceq y$, which by antisymmetry of $\preceq$ force $x = x'$ and $y = y'$. For the lexicographic order it is a bit more tricky: if we have $(x, y) \preceq_L (x', y') \preceq_L (x, y)$ then either

(i) $x \prec x' \prec x$ (impossible by antisymmetry of $\preceq$ and definition of $\prec$),

(ii) $x = x'$, $y \preceq y'$, and $x' \prec x$ (impossible by definition of $\prec$),

(iii) $x \prec x'$, $x' = x$, and $y' \preceq y$ (impossible by definition of $\prec$),

(iv) $x = x'$, $y \preceq y'$, $x' = x$, $y' \preceq y$, which force $x = x'$ and $y = y'$.

In the only possible case, we have $(x, y) = (x', y')$, showing that $\preceq_L$ is antisymmetric.

The other parts of this proof are left for exercises. ■

The same constructions extend to any product $A^n$, in the obvious way:

$$\begin{aligned}(x_1, \ldots, x_n) \preceq_L (y_1, \ldots, y_n) \quad \Leftrightarrow \quad & x_1 \prec y_1 \text{ or} \\ & (x_1 = y_1 \text{ and } x_2 \prec y_2) \text{ or} \\ & (x_1 = y_1 \text{ and } x_2 = y_2 \text{ and } x_3 \prec y_3) \text{ or} \\ & \cdots \text{ or} \\ & (x_1 = y_1 \text{ and } x_2 = y_2 \text{ and } \cdots x_n \preceq y_n)\end{aligned}$$

$$(x_1, \ldots, x_n) \preceq_P (y_1, \ldots, y_n) \;\Leftrightarrow\; x_1 \preceq y_1 \text{ and } x_2 \preceq y_2 \text{ and } \cdots \text{ and } x_n \preceq y_n$$

The lexicographic order can also apply to finite **words** (words are ordered lists of variable length) in which case it still corresponds to "dictionary order", but finite words are not within our syllabus.

Finally, the same constructions can be applied to any cartesian product $A \times B$, even if $A$ and $B$ have different orders $\preceq_A$ and $\preceq_B$. In that case, for example, $(a, b) \preceq_L (a', b')$ when $a \prec_A a'$, or $a = a'$ and $b \preceq_B b'$.

## 8.3 Drawing Orders

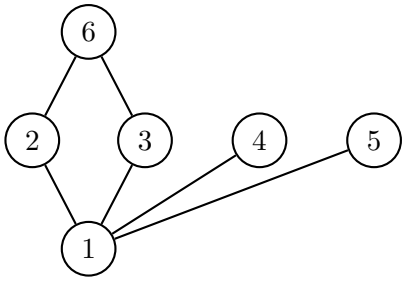
Since an order is an example of a relation, we can draw it as a directed graph, just like in Section 4.5. But there is a more concise and attractive way to draw partial orders and linear orders, in which the reflexivity and transitivity of the order are implicit. A **Hasse diagram** for a partial order $\preceq$ on a set $A$ is a graph, drawn in the plane, with vertices corresponding to the elements of $A$ and an edge going *up* from $a$ to $b$ if $a \prec b$ (so excluding $a = b$) and there is no element $x$ with $a \prec x \prec b$.

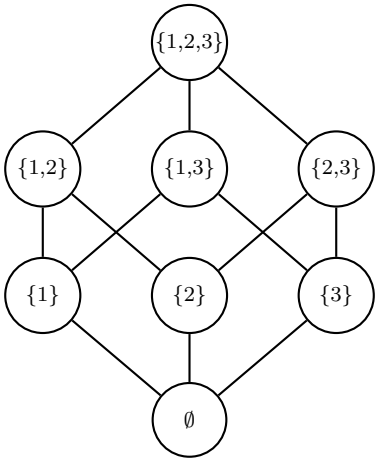
Because of the particular orientation requirement, edges do not need arrows (they always "point" upwards). The definition specifically excludes loop edges because reflexivity is automatic for all partial orders, and unnecessary edges which can be deduced from transitivity. As a result, a Hasse diagram of a partial order is much easier to read than the full digraph containing all the edges. (In general, the relation which results from deleting loops, and edges redundant under transitivity, is called the **cover relation**.)

If $A$ is an infinite set, the complete Hasse diagram cannot, of course, be drawn. But when the order relation follows a pattern, drawing an indicative part of the Hasse diagram can be helpful.

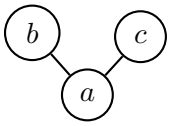
The simplest Hasse diagrams are of linear orders: they are simply ascending sequences of nodes. Here are two examples of diagrams of partial orders. On the left, the set $\{1, 2, 3, 4, 5, 6\}$ ordered by divisibility. Note that the edge $1 \,|\, 6$, and all $n \,|\, n$, are not drawn. On the right, $\mathcal{P}(\{1, 2, 3\})$ ordered by $\subseteq$:

the diagram reflects the symmetrical nature of this partial order.

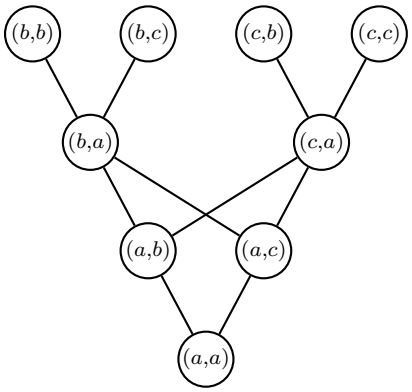


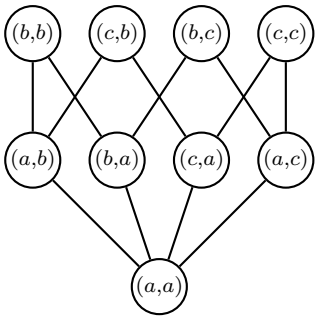

Let us use Hasse diagrams to illustrate the distinction between lexicographic and product order. Take the set $\{a, b, c\}$, with order




That is, $a \preceq b$ and $a \preceq c$, but $b$ and $c$ are incomparable. Then the corresponding lexicographic and product orders on $\{a, b, c\}^2$ are given by:




and



There are often many different ways to place the nodes in a Hasse diagram, and some layouts are more informative than others in that they reflect underlying symmetries in the order, while trying to reduce confusing line-crossing. Finding good layouts is something of an art and it is difficult to automate.

## 8.4 Upper and Lower Bounds

In a partial or linear order it makes sense to consider some elements to be greater than others (it does not really make sense to do so for a preorder which is not antisymmetric).

**Definition** Suppose that $\preceq$ is a partial order on $A$ and $S \subseteq A$.

Then we say that $m$ is an **upper bound** for $S$ if $x \preceq m$ for all $x \in S$. $m$ is a **lower bound** for $S$ if $m \preceq x$ for all $x \in S$.

We say that $m$ is a **maximum** of $S$ if $x \preceq m$ for all $x \in S$, and also $m \in S$. Similarly, $m$ is a **minimum** of $S$ is $m \preceq x$ for all $x \in S$, and also $m \in S$.

Another way to say that $m$ is an upper bound for $S$ is to say that $S$ is **bounded above** by $m$; in some circumstances we might say that $S$ is **dominated** by $m$.

Consider the linear order $\leq$ on $\mathbb{R}$. Then 1 is an upper bound for $(0, 1)$ but it is *not* a maximum because it is not a member of the set. Also, 2 is an upper bound for $(0, 1)$: upper (and lower) bounds are not necessarily unique. On the other hand, if a set has a maximum, it is unique (proof: exercise). Or consider the partial order $\mid$ on $\mathbb{N}_+$; the upper bounds for the set $\{3, 4, 5\}$ are the numbers divisible by all of 3, 4 and 5 (so for example 60 is an upper bound) and the lower bounds are the numbers which divide into all of 3, 4 and 5 (the only such number is 1).

It is possible to get very confused if the order itself is counterintuitive. For example, with respect to the order $\geq$ on $\mathbb{R}$, 0 is an *upper* bound and the maximum for $[0, 1]$, and 1 is a *lower* bound and the minimum...

Note that sets need not have an upper or lower bound. For example, $[0, \infty)$ in the linear order $\leq$ has lower bounds, but no upper bound. Or consider the set of English words ordered by prefix: there is no lower or upper bound

(no English word is a prefix of every other, nor begins with every other as a prefix).

Also, a set might have an upper bound (or more than one) but no maximum. This is true of $(0,1)$, which has upper bounds (anything $\geq 1$) and lower bounds (anything $\leq 0$) but no maximum or minimum: anything greater than (or less than) every member of $(0,1)$ cannot itself be a member of $(0,1)$. But 0 and 1 are the tightest possible lower and upper bounds: that leads to the following definition.

**Definition** Suppose that $\preceq$ is a partial order on $A$ and $S \subseteq A$.

Then we say that $m$ is a **least upper bound** for $S$ if $m \in A$ is an upper bound for $S$, and any other upper bound for $S$, $m'$, satisfies $m \preceq m'$. We write this $m = \operatorname{lub} S$.

We say that $m$ is a **greatest lower bound** for $S$ if $m \in A$ is a lower bound for $S$, and any other lower bound for $S$, $m'$, satisfies $m' \preceq m$. We write this $m = \operatorname{glb} S$.

Mathematicians often refer to least upper bound as **supremum** and greatest lower bound as **infimum**. In lattice theory (the wider study of orders) they are sometimes called **join** and **meet**. We will stick with the abbreviations **lub** and **glb**.

So 0 and 1 are the glb and lub of $(0,1)$ with respect to the $\leq$ order. If we consider the order $|$ on $\mathbb{N}_+$, $\operatorname{lub}\{2,3\}$ is 6. This is because all upper bounds of $\{2,3\}$ must be divisible by both 2 and 3, so the set of upper bounds is $\{6, 12, 18, \ldots\}$ and 6 is the least upper bound. Remember that *least*, here, is still with respect to the $|$ order.

Be careful! If a set has a lub it must be unique (similarly for glb) because if $m$ and $m'$ are both lubs for $S$ then $m \leq m'$ and $m' \leq m$, and (remember we are only defining lubs and glbs for partial orders) antisymmetry forces $m = m'$. But not every subset has a lub or glb. For a start, a set with no upper bound cannot have a least upper bound (e.g. $(0, \infty)$ as a subset of the linear order $\leq$ on $\mathbb{R}$). But, for a more serious example, take the order $\leq$ on $\mathbb{Q}$, and consider the subset $\mathbb{Q} \cap (0, \sqrt{2})$. The upper bounds for this set are rationals $q > \sqrt{2}$, and we will show in the next section that there is no least upper bound: $\sqrt{2}$ itself can't be a least upper bound, because $\sqrt{2} \notin \mathbb{Q}$.

Remember that lub $S$ has to be a member of the order in which it lives, even if it is not a member of $S$.

When every *pair* of elements of $A$ has a lub and glb, we say that the order is a **lattice**; then the lub and glb operations become binary operators, and they are often written infix as $x \sqcup y$ and $x \sqcap y$ respectively. When every subset of $A$ has a lub and glb, we say that the order is a **complete lattice**, but this takes us beyond the syllabus. It is notable that $\mathbb{Q}$, ordered by $\leq$, is not a complete lattice but $\mathbb{R}$, ordered by $\leq$, is a complete lattice. This is the fundamental distinction between $\mathbb{Q}$ and $\mathbb{R}$.

It can be awkward to prove that $m$ is the least upper bound of $S$, because we have to reason about every possible upper bound for $S$. When $A$ is a **linear order**, there is an alternative definition which is sometimes more convenient:

**Claim 8.2** Let $\preceq$ be a linear order on a set $A$ and $S \subseteq A$. Then $m = \text{lub}\, S$ if and only if

(i) $x \preceq m$ for all $x \in S$, and

(ii) for all $a \in A$, if $a \prec m$ then there exists an element $x \in S$ with $a \prec x$.

Note that this is only equivalent for linear orders: it is equivalent because $a \prec m \Rightarrow \exists x.a \prec x$ is the contrapositive of the statement $\forall x.x \preceq a \Rightarrow m \preceq a$, when we can interchange $x \not\preceq x'$ with $x' \prec x$. The alternative definition of glb is symmetrical.

This definition makes it easier to prove that, for example, 1 is the lub of $(0, 1)$ with respect to the linear order $\leq$: all $x \in (0, 1)$ satisfy $x \leq 1$, and if $a < 1$ then $\max(\frac{1}{2}, \frac{1+a}{2})$ is an element of $(0, 1)$ greater than $a$.

## 8.5 Proving Sentences of the form $\forall x.\exists y.P$

The statement "$S$ has an upper bound" is of the form $\exists x.\forall y.P$: **there exists** an $x$ (the upper bound) such that, **for all** $y \in S$, $y \preceq x$. We saw how to construct proofs of these statements in Chapter 7.

The statement "$S$ has no upper bound" can be written in the form $\forall x.\exists y.P$: **for all** $x \in A$, **there exists** some $y \in S$ with $x \not\preceq y$. The second part of alternative definition of $m = \text{lub}\, S$ for linear orders is also of this form. The same

pattern is found in "$f$ is onto" (**for all** members $x$ of the codomain, **there exists** a member of the domain which maps to $x$), or "$f(n) \neq O(g(n))$" (**for all** $N$ and $c$, **there exists** $n \geq N$ with $|f(n)| > c|g(n)|$).

We now briefly consider how to prove statements of this form. We must imagine that we are *given* a value of $x$, over which we have no control, and are required to *find* a value of $y$ so that $P$ becomes true: the value of $y$ will probably depend on the $x$ we are given. In effect, we are constructing a function which produces a good value of $y$ (to make $P$ true) out of any value of $x$, although we do not need to give an explicit formula for the function. (These two concepts – seeing quantifiers as "give" and "take", and converting $\forall x.\exists y.P$ into the existence of a function – are interesting topics in the theory of logic, but are not part of the discrete mathematics course.)

First, a very simple example.

**Claim 8.3** The set $\mathbb{N}$, ordered by $|$, has no upper bound.

**Proof** Intuitively, it seems obvious. Formally, we must show that any $n \in \mathbb{N}$ is not an upper bound: for any $n \in \mathbb{N}$, there exists $n' \in \mathbb{N}$ with $n' \not| \, n$. (Imagine that we are *given* $n$, we need to *find* a corresponding value of $n'$ with that property.)

Given any $n \in \mathbb{N}$, we can choose $n' = n + 1$. Then, as we hoped, $n' \not| \, n$. ■

Now let us prove something we mentioned in the previous section.

**Claim 8.4** The set $S = \mathbb{Q} \cap (0, \sqrt{2})$, as a subset of the order $\mathbb{Q}$ ordered by $\leq$, has no least upper bound.

**Proof** This is a proof by contradiction: we suppose that $m = \operatorname{lub} S$ exists and derive a contradiction by showing that there is a rational $m' < m$ which is also an upper bound for $S$ (it is a contradiction because, by definition, $m$ is supposed to be the *least* upper bound). Again, this is of the form: *given* a supposed lub $m$, we must *find* an upper bound which is less.

Remember that $m = \operatorname{lub} S$ means $m \in \mathbb{Q}$: the lub has to be a member of the ordered set. So let us write $m = p/q$, for integers $p$ and $q$. To complete the contradiction we need to find a rational $m'$ which satisfies

$$\sqrt{2} \leq m' < m$$

(the first inequality tell us that $m'$ is an upper bound for $S$, and the second that it is less than $m$). Since $m$ is an upper bound for $S$, and $\sqrt{2} \notin \mathbb{Q}$, we know that $m - \sqrt{2}$ is positive. So let us write $r = \lceil 1/(m - \sqrt{2}) \rceil$; we know $r \in \mathbb{N}_+$ so the following $m'$ will do:

$$m' = \frac{\lceil r\sqrt{2} \rceil}{r}.$$

Why does this $m'$ work? First, since $\lceil x \rceil \geq x$, we have $m' \geq \sqrt{2}$. Second, because $\lceil x \rceil < x + 1$, we have $m' < \sqrt{2} + 1/r \leq \sqrt{2} + m - \sqrt{2} = m$. ■

In this proof, we had to take care to construct the right value of $m'$. We used the fact that any rational approximation to an irrational (here $m$ as an upper bound for $\sqrt{2}$) can always be made closer if we use a rational with a large enough denominator.

## 8.6 Interesting Diversion: Order Isomorphisms

Let $A$ and $B$ be sets, and suppose that $f : A \to B$ is a bijection. If $A$ and $B$ have order relations $\preceq_A$ and $\preceq_B$, respectively, it makes sense to compare the statements $a \preceq_A a'$ and $f(a) \preceq_B f(a')$. When they are the same, $f$ is called an **order isomorphism**.

**Definition** If $A$ and $B$ have order relations $\preceq_A$ and $\preceq_B$, an **order isomorphism** between $A$ and $B$ is a bijection $f : A \to B$ such that

$$a \preceq_A a' \quad \Leftrightarrow \quad f(a) \preceq_B f(a').$$

Let us write $(A, \preceq_A)$ so to identify concisely both a set an the order relation on it. There is an order isomorphism between $(\mathbb{R}, \leq)$ and $(\mathbb{R}, \geq)$ given by $f : x \mapsto -x$; this is a very simple example. An order isomorphism between $(\mathbb{N}, \leq)$ and $(\{1, 2, 4, 8, 16, \ldots\}, |\,)$ is given by $x \mapsto 2^x$. If we order English words by prefix, there is an order isomorphism between {"ox", "oxford", "oxfordshire", "oxen"} and $(\{1, 2, 3, 4\}, |\,)$: identify 1 with "ox", 2 with "oxford", 3 with "oxen", and 4 with "oxfordshire". Perhaps even more obscurely, if we take the vertices of a cube, label one vertex the origin $O$ and say that vertex $A$ is less than vertex $B$ if a shortest path from $O$ to $B$ includes $A$, we create a partial order on the 8 vertices. It can be seen

that there is an order isomorphism between this order and $(\mathcal{P}(\{1,2,3\}),\subseteq)$. The best way to see this is to draw Hasse diagrams of both orders.

Just as the existence of a bijection between $A$ and $B$ forces them to have to same cardinality, so the existence of an order isomorphism between $(A,\preceq_A)$ and $(B,\preceq_B)$ forces them to have some of the same attributes: if one is a partial order, so is the other, similarly linear order, lattice, etc.

---

## Practice Questions

**8.1** Consider the order $\subseteq$ on the set $\mathcal{P}(\mathbb{Z}_{10})$. Which of the following sets are chains, and which are antichains?

(i) $\emptyset$,
(ii) $\{\{1\}, \{1,3,5\}, \{1,3\}, \{1,3,5,9\}\}$,
(iii) $\{\{1,2\}, \{2,3\}, \{3,4\}\}$,
(iv) $\{\{3,1\}, \{4\}, \{1,5,9\}, \{2,6,4,5\}\}$.

**8.2** Give an example of a linear order $\preceq$ on a set $A$ such that the product order $\preceq_P$ on $A \times A$ is *not* a linear order.

**8.3** Let $A$ be a partial order. Prove that if $S \subseteq A$ has a maximum then it is unique.

**8.4** With respect to the partial order $\mid$ on the set $\mathbb{N}_+$, find

(i) $\text{lub}\{4,6\}$,
(ii) $\text{lub}\{3,4,5,6\}$,
(iii) $\text{glb}\{12,16,18,24\}$,
(iv) $\text{glb}\{737,2345\}$.

**8.5** Consider the set $A = \{B \subseteq \{1,2,3,4\} \mid |B| \neq 2\}$, ordered by $\subseteq$. Draw the Hasse diagram of this order, and find one pair of elements with no lub, and one pair of elements with no glb.

**8.6** Consider the set $\mathbb{N}^2$, with the lexicographic order formed from $\leq$. Define $S = \text{lub}\{(0,n) \mid n \in \mathbb{N}\}$. Show that $S$ has no maximum, and find $\text{lub}\, S$.

**8.7** Let $A \subseteq B$ be subsets of some set, which has a partial order $\leq$. Prove that, if $\text{lub}\, A$ and $\text{lub}\, B$ both exist, then $\text{lub}\, A \leq \text{lub}\, B$.

**8.8** Find an order isomorphism between the set of positive integer divisors of 385, ordered by $\mid$, and $\mathcal{P}(\{a,b,c\})$, ordered by inclusion.

## Answers to Chapter 8 Practice Questions

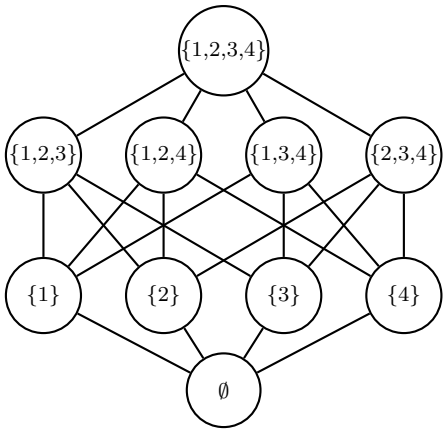
**8.1** (i) vacuously, both a chain and an antichain; (ii) a chain; (iii) an antichain; (iv) neither chain nor antichain.

**8.2** There are many examples, including $\leq$ on $\mathbb{N}$.

**8.3** Let $m$ and $m'$ be maxima for $S$. Since $m \in S$ we must have $m \preceq m'$; since $m' \in S$ we must have $m' \preceq m$. Since we are working in a partial order (antisymmetric) we have $m = m'$.

**8.4** (i) The upper bounds divide 4 and 6, i.e. they divide 12. The least, with respect to the divides order, is 12. (ii) Similarly, $\text{lub}\{3, 4, 5, 6\}$ is the least common multiple 60. (iii) The lower bounds are those which are divisors of 12, 16, 18, and 24 i.e. $\{1, 2\}$, so the glb is 2. (iv) Similarly, $\text{glb}\{737, 2345\}$ is the greatest common divisor 67 (use Euclid's algorithm).

**8.5**




Pairs such as $\{1\}$ and $\{2\}$ have no lub and pairs such as $\{1, 2, 3\}$ and $\{1, 2, 4\}$ have no glb.

**8.6** The only possibilities for a maximum are $(0, n)$ with $n \in \mathbb{N}$. But none of these can be an upper bound, because $(0, n + 1)$ is always greater (with respect to the lexicographic order). $\text{lub}\, S = (1, 0)$.

**8.7** For any $a \in A$, we have $a \in B$; then $a \leq \text{lub}\, B$ because $\text{lub}\, B$ is an upper bound for $B$. We have proved that $\text{lub}\, B$ is an upper bound for $A$. But $\text{lub}\, A$ is the least upper bound for $A$, so $\text{lub}\, A \leq \text{lub}\, B$.

**8.8** It might be helpful to draw a picture. There are 6 different (symmetrical) order isomorphisms, including $\emptyset \leftrightarrow 1$, $\{a\} \leftrightarrow 5$, $\{b\} \leftrightarrow 7$, $\{c\} \leftrightarrow 11$, $\{a, b\} \leftrightarrow 35$, $\{b, c\} \leftrightarrow 77$, $\{a, c\} \leftrightarrow 55$, $\{a, b, c\} \leftrightarrow 385$.

# Index