CHAPTER 2

Functions

Introduction

Functions play a fundamental role in nearly all of mathematics. Combinatorics is no exception. In the next section we review the basic terminology and notation for functions. Permutations are special functions that arise in a variety of ways in combinatorics. Besides studying them for their own interest, we'll see them as a central tool in counting objects with symmetries and as a source of examples for decision trees in later chapters. Section 3 discusses some combinatorial aspects of functions that relate to some of the material in the previous chapter. We conclude the chapter with a discussion of Boolean functions. They form the mathematical basis of most computer logic.

2.1 Some Basic Terminology

Terminology for Sets

Except for the real numbers ($\mathbb{R}$), rational numbers ($\mathbb{Q}$) and integers ($\mathbb{Z}$), our sets are normally finite. The set of the first n positive integers, $\{1, 2, \ldots, n\}$ will be denoted by $\underline{n}$.

Recall that $|A|$ is the number of elements in the set A. When it is convenient to do so, we'll assume that the elements of a set A have been linearly ordered and denote the ordering by $a_1, a_2, \ldots, a_{|A|}$. Unless clearly stated otherwise, the ordering on a set of numbers is the numerical ordering. For example, the ordering on $\underline{n}$ is $1, 2, 3, \ldots, n$.

If A and B are sets, we write $A - B$ for the set of elements in A that are not in B:

$$A - B \;=\; \{\, x \mid x \in A \text{ and } x \notin B \,\}.$$

(This is also written $A \backslash B$.)

If A and B are sets, recall from the previous chapter that the Cartesian product $A \times B$ is the set of all ordered pairs built from A and B:

$$A \times B \;=\; \{\, (a, b) \mid a \in A \text{ and } \; b \in B \,\}.$$

We also call $A \times B$ the *direct product* of A and B.

If $A = B = \mathbb{R}$, the real numbers, then $\mathbb{R} \times \mathbb{R}$, written $\mathbb{R}^2$ is frequently interpreted as coordinates of points in the plane. Two points are the same if and only if they have the same coordinates, which says the same thing as our definition of $(a, b) = (a', b')$. Recall that the direct product can be extended to any number of sets. How can $\mathbb{R} \times \mathbb{R} \times \mathbb{R} = \mathbb{R}^3$ be interpreted?

What are Functions?

Definition 2.1 Function *If A and B are sets, a* **function** *from A to B is a rule that tells us how to find a unique $b \in B$ for each $a \in A$. We write $f\colon A \to B$ to indicate that f is a function from A to B. We call the set A the* **domain** *of f and the set B the* **codomain** *of f. To specify a function completely you must give its domain, codomain and rule. The set of all functions from A to B is written B^A, for a reason we will soon explain. Thus $f\colon A \to B$ and $f \in B^A$ say the same thing.*

In calculus you dealt with functions whose codomains were $\mathbb{R}$ and whose domains were contained in $\mathbb{R}$; for example, $f(x) = 1/(x^2-1)$ is a function from $\mathbb{R} - \{-1, 1\}$ to $\mathbb{R}$. You also studied functions of functions! The derivative is a function whose domain is all differentiable functions and whose codomain is all functions. If we wanted to use functional notation we could write $D(f)$ to indicate the function that the derivative associates with f. Can you see how to think of the integral as a function? This is a bit tricky because of the constant of integration. We won't pursue it.

Definition 2.2 One-line notation *When A is ordered, a function can be written in* **one-line notation** *as $(f(a_1), f(a_2), \ldots, f(a_{|A|}))$. Thus we can think of function as an element of $B \times B \times \ldots \times B$, where there are $|A|$ copies of B. Instead of writing $B^{|A|}$ to indicate the set of all functions, we write B^A. Writing $B^{|A|}$ is incomplete because the domain is not specified. Instead, only its size is given.*

Example 2.1 Using the notation To get a feeling for the notation used to specify a function, it may be helpful to imagine that you have an envelope or box that contains a function. In other words, this envelope contains all the information needed to completely describe the function. Think about what you're going to see when you open the envelope.

* * * Stop and think about this! * * *

You might see

$$P = \{a, b, c\}, \qquad g\colon P \to \underline{4}, \qquad g(a) = 3, \quad g(b) = 1 \quad \text{and} \quad g(c) = 4.$$

This tells you that the name of the function is g, the domain of g is P, which is $\{a, b, c\}$, and the codomain of g is $\underline{4} = \{1, 2, 3, 4\}$. It also tells you the values in $\underline{4}$ that g assigns to each of the values in its domain. Someone else may have put

$$g\colon \{a, b, c\} \to \underline{4}, \qquad \text{ordering: } a, b, c, \qquad g = (3, 1, 4).$$

in the envelope instead. This describes the same function but doesn't give a name for the domain. On the other hand, it gives an order on the domain so that the function can be given in one line form. Can you describe other possible envelopes for the same function?

What if the envelope contained only $g = (3, 1, 4)$? You've been cheated! You *must* know the domain of g in order to known what g is. What if the envelope contained

$$\text{the domain of } g \text{ is } \{a, b, c\}, \qquad \text{ordering: } a, b, c, \qquad g = (3, 1, 4)?$$

We haven't specified the codomain of g, but is it necessary since we know the values of the function? Our definition included the requirement that the codomain be specified, so this is not a complete definition. On the other hand, we frequently only need to know which values in its codomain g actually takes on (here 1, 3 and 4), so we'll be sloppy in such cases and accept this as if it were a complete specification. □

Example 2.2 Counting functions By the Rule of Product, $\left|B^A\right| = |B|^{|A|}$. We can represent a subset S of A by a unique function $f: A \to \underline{2}$ where $f(x) = 1$ if $a \notin S$ and $f(x) = 2$ if $x \in S$. This proves that there are $2^{|S|}$ such subsets. We can represent a list of k elements of a set S with repetition allowed by a unique function $f: \underline{k} \to S$. In this representation, the list corresponds to the function written in one line notation. (Recall that the ordering on $\underline{k}$ is the numerical ordering.) This proves that there are exactly $|S|^k$ such lists. □

Definition 2.3 Types of functions *Let $f: A \to B$ be a function. If for every $b \in B$ there is an $a \in A$ such that $f(a) = b$, then f is called a* **surjection** *(or an* **onto function***). Another way to describe a surjection is to say that it takes on each value in its codomain at least once.*

If $f(x) = f(y)$ implies $x = y$, then f is called an **injection** *(or a* **one-to-one function***). Another way to describe an injection is to say that it takes on each value in its codomain at most once. The injections in $S^{\underline{k}}$ correspond to lists without repetitions.*

If f is both an injection and a surjection, it is a called a **bijection***. The bijections of A^A are called the* **permutations** *of A. If $f: A \to B$ is a bijection, we may talk about the* **inverse** *of f, written f^{-1}, which reverses what f does. Thus $f^{-1}: B \to A$ and $f^{-1}(b)$ is that unique $a \in A$ such that $f(a) = b$. Note that $f(f^{-1}(b)) = b$ and $f^{-1}(f(a)) = a$. Do not confuse f^{-1} with $1/f$. For example, if $f: \mathbb{R} \to \mathbb{R}$ is given by $f(x) = x^3+1$, then $1/f(x) = 1/(x^3+1)$ and $f^{-1}(y) = (y-1)^{1/3}$.*

Example 2.3 Using the notation We'll illustrate the ideas in the previous paragraph. Let $A = \underline{4}$, $B = \{a, b, c, d, e\}$ and $f = (d, c, d, a)$. Since the value d is taken on twice by f, f is not an injection. Since the value b is not taken on by f, f is not a surjection. (We could have said e is not taken on, instead.) The function (b, d, c, e) is an injection since there are no repeats in the list of values taken on by the function.

Now let $A = \underline{4}$, $B = \{x, y, z\}$ and $g = (x, y, x, z)$. Since every element of B appears at least once in the list of values taken on, f is a surjection.

Finally, let $A = B = \underline{4}$ and $h = (3, 1, 4, 2)$. The function is both an injection and a surjection. Hence, it is a bijection. Since the domain and codomain are the same and f is a bijection, it is a permutation of $\underline{4}$. The inverse of h is $(2, 4, 1, 3)$. □

Example 2.4 Two-line notation Since one line notation is a simple, brief way to specify functions, we'll use it frequently. If the domain is not a set of numbers, the notation is poor because we must first pause and order the domain. There are other ways to write functions which overcome this problem. For example, we could write $f(a) = 4$, $f(b) = 3$, $f(c) = 4$ and $f(d) = 1$. This could be shortened up somewhat to $a \to 4$, $b \to 3$, $c \to 4$ and $d \to 1$. By turning each of these sideways, we can shorten it even more: $\left(\begin{smallmatrix} a & b & c & d \\ 4 & 3 & 4 & 1 \end{smallmatrix}\right)$ For obvious reasons, this is called *two-line notation*. Since x always appears directly over $f(x)$, there is no need to order the domain; in fact, we need not even specify the domain separately since it is given by the top line. If the function is a bijection, its inverse is obtained by interchanging the top and bottom lines.

The arrows we introduced in the last paragraph can be used to help visualize different properties of functions. Imagine that you've listed the elements of the domain A in one column and the elements of the codomain B in another column to the right of the the domain. Draw an arrow from a to b if $f(a) = b$. Thus the heads of arrows are labeled with elements of B and the tails with elements of A. Since f is a function, no two arrows have the same tail. If f is an injection, no two arrows have the same head. If f is a surjection, every element of B is on the head of some arrow. You should be able to describe the situation when f is a bijection. □

Exercises

2.1.1. This exercise lets you check your understanding of the definitions. In each case below, some information about a function is given to you. Answer the following questions and give reasons for your answers: (The answers are given at the end of this problem set.)

(i) Have you been given enough information to specify the function; i.e., would this be enough data for a function envelope?

(ii) Can you tell whether or not the function is an injection? a surjection? a bijection? If so, what is it?

(iii) If possible, give the function in two line form.

(a) $f \in \underline{3}^{\{\heartsuit,\clubsuit,\diamondsuit,\spadesuit\}}$, $\quad f = (3, 1, 2, 3)$.

(b) $f \in \{\heartsuit, \clubsuit, \diamondsuit, \spadesuit\}^{\underline{3}}$, $\quad f = (\spadesuit, \heartsuit, \clubsuit)$.

(c) $f \in \underline{4}^{\underline{3}}$, $\quad 2 \to 3, \quad 1 \to 4, \quad 3 \to 2$.

2.1.2. Let A and B be finite sets and $f\colon A \to B$. Prove the following claims. Some are practically restatements the of definitions, some require a few steps.

(a) If f is an injection, then $|A| \leq |B|$.

(b) If f is a surjection, then $|A| \geq |B|$.

(c) If f is a bijection, then $|A| = |B|$.

(d) If $|A| = |B|$, then f is an injection if and only if it is a surjection.

(e) If $|A| = |B|$, then f is a bijection if and only if it is an injection or it is a surjection.

Answers

2.1.1. (a) We know the domain and codomain of f. By Exercise 2, f cannot be an injection. Since no order is given for the domain, the attempt to specify f in one-line notation is meaningless. If the attempt at specification makes any sense, it tells us that f is a surjection. We cannot give it in two line form since we don't know the function.

(b) We know the domain and codomain of f and the domain has an implicit order. Thus the one-line notation specifies f. It is an injection but not a surjection. In two line form it is $\left(\begin{smallmatrix} 1 & 2 & 3 \\ \spadesuit & \heartsuit & \clubsuit \end{smallmatrix}\right)$.

(c) This function is specified and is an injection. In one-line notation it would be (4,3,2), and, in two line notation, $\left(\begin{smallmatrix} 1 & 2 & 3 \\ 4 & 3 & 2 \end{smallmatrix}\right)$.

2.2 Permutations

Before beginning our discussion, we need the notion of composition of functions. Suppose that f and g are two functions such that the values f takes on are contained in the domain of g. We can write this as $f\colon A \to B$ and $g\colon C \to D$ where $f(a) \in C$ for all $a \in A$. We define the *composition* of g and f, written $gf\colon A \to D$ by $(gf)(x) = g(f(x))$ for all $x \in A$. The notation $g \circ f$ is also used to denote composition. Suppose that f and g are given in two line notation by

$$f = \begin{pmatrix} p & q & r & s \\ P & R & T & U \end{pmatrix} \qquad g = \begin{pmatrix} P & Q & R & S & T & U & V \\ 1 & 3 & 5 & 2 & 4 & 6 & 7 \end{pmatrix}.$$

Then $gf = \begin{pmatrix} p & q & r & s \\ 1 & 5 & 4 & 6 \end{pmatrix}$.

The set of permutations on a set A is denoted in various ways in the literature. Two notations are $\mathrm{PER}(A)$ and $\mathcal{S}(A)$. Suppose that f and g are permutations of a set A. Recall that a permutation is a bijection from a set to itself and so it makes sense to talk about f^{-1} and fg. We claim that fg and f^{-1} are also permutations of A. This is easy to see if you write the permutations in two-line form and note that the second line is a rearrangement of the first if and only if the function is a permutation.

Again suppose that f is a permutation. Instead of $f \circ f$ or ff we write f^2. Note that $f^2(x)$ is not $(f(x))^2$. (In fact, if multiplication is not defined in A, $(f(x))^2$ has no meaning.) We could compose three copies of f. The result is written f^3. In general, we can compose k copies of f to obtain f^k. A cautious reader may be concerned that $f \circ (f \circ f)$ may not be the same as $(f \circ f) \circ f$. They are equal. In fact, $f^{k+m} = f^k \circ f^m$ for all nonnegative integers k and m, where f^0 is defined by $f^0(x) = x$ for all x in the domain. This is based on the "associative law" which states that $f \circ (g \circ h) = (f \circ g) \circ h$ whenever the compositions make sense. We'll prove these results.

To prove that the two functions are equal, it suffices to prove that they take on the same values for all x in the domain. Let's use this idea for $f \circ (g \circ h)$ and $(f \circ g) \circ h$. We have

$$\begin{aligned} (f \circ (g \circ h))(x) &= f((g \circ h)(x)) && \text{by the definition of } \circ, \\ &= f(g(h(x))) && \text{by the definition of } \circ. \end{aligned}$$

Similarly

$$\begin{aligned} ((f \circ g) \circ h)(x) &= (f \circ g)(h(x)) && \text{by the definition of } \circ, \\ &= f(g(h(x))) && \text{by the definition of } \circ. \end{aligned}$$

More generally, one can use this approach to prove by induction that $f_1 \circ f_2 \circ \cdots \circ f_n$ is well defined. This result then implies that $f^{k+m} = f^k \circ f^m$. Note that we have proved that the associative law for any three functions f, g and h for which the domain of f contains the values taken on by g and the domain of g contains the values taken on by h.

Example 2.5 Using two-line and composition notations Let f and g be the permutations

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 1 & 4 & 5 & 3 \end{pmatrix} \qquad g = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 1 \end{pmatrix}.$$

We can compute fg by calculating all the values. This can be done fairly easily from the two line form: For example, $(fg)(1)$ can be found by noting that the image of 1 under g is 2 and the image of 2 under f is 1. Thus $(fg)(1) = 1$. You should be able to verify that

$$fg = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 4 & 5 & 3 & 2 \end{pmatrix} \quad \text{and} \quad gf = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 2 & 5 & 1 & 4 \end{pmatrix} \neq fg$$

and that

$$f^2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 5 & 3 & 4 \end{pmatrix} \quad f^3 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 1 & 3 & 4 & 5 \end{pmatrix} \quad \text{and} \quad g^5 = f^6 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix}.$$

Note that it is easy to get the inverse, simply interchange the two lines. Thus

$$f^{-1} = \begin{pmatrix} 2 & 1 & 4 & 5 & 3 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix} \quad \text{which is the same as } f^{-1} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 1 & 5 & 3 & 4 \end{pmatrix},$$

since the order of the columns in two line form does not matter. □

Let f be a permutation of the set A and let $n = |A|$. If $x \in A$, we can look at the sequence $x, f(x), f(f(x)), \ldots, f^k(x), \ldots$, which is often written as $x \to f(x) \to f(f(x)) \to \cdots \to f^k(x) \to \cdots$. Since the codomain of f has n elements, this sequence will contain a repeated element in the first $n+1$ entries. Suppose that $f^s(x)$ is the first sequence entry that is ever repeated and that $f^{s+p}(x)$ is the first time that it is repeated. Apply $(f^{-1})^s$ to both sides of this equality to obtain $x = f^p(x)$ and so, in fact, $s = 0$. It follows that the sequence cycles through a pattern of length p forever since $f^{p+1}(x) = f(f^p(x)) = f(x)$, $f^{p+2}(x) = f^2(f^p(x)) = f^2(x)$, and so on. We call $(x, f(x), \ldots, f^{p-1}(x))$ the *cycle* containing x and call p the *length of the cycle*. If a cycle has length p, we call it a p-cycle. Cyclic shifts of a cycle are considered the same; for example, if (1,2,6,3) is the cycle containing 1 (as well as 2, 3 and 6), then (2,6,3,1), (6,3,1,2) and (3,1,2,6) are other ways of writing the cycle.

Example 2.6 Using cycle notation Consider the permutation $f = \left(\begin{smallmatrix} 1&2&3&4&5&6&7&8&9 \\ 2&4&8&1&5&9&3&7&6 \end{smallmatrix}\right)$. Since $1 \to 2 \to 4 \to 1$, the cycle containing 1 is (1,2,4). We could equally well write it (2,4,1) or (4,1,2); however, (1,4,2) is different since it corresponds to $1 \to 4 \to 2 \to 1$ The usual convention is to list the cycle starting with its smallest element. The cycles of f are (1,2,4), (3,8,7), (5) and (6,9). We write f in *cycle form* as

$$f = (1,2,4)\ (3,8,7)\ (5)\ (6,9).$$

It is common practice to omit the cycles of length one and write $f = (1,2,4)(3,8,7)(6,9)$. The inverse of f is obtained by reading the cycles backwards because $f^{-1}(x)$ is the lefthand neighbor of x in a cycle. Thus

$$f^{-1} = (4,2,1)(7,8,3)(9,6) = (1,4,2)(3,7,8)(6,9). \quad \square$$

Cycle form is useful in certain aspects of the branch of mathematics called "finite group theory." We will find it useful later when we study the problem of counting structures having symmetries. Here's an application now.

Example 2.7 Powers of permutations With a permutation in cycle form, its very easy to calculate a power of the permutation. For example, suppose we want the tenth power of the permutation whose cycle form (including cycles of length 1) is $(1,5,3)(7)(2,6)$. To find the image of 1, we take ten steps: $1 \to 5 \to 3 \to 1 \cdots$. Where does it stop after ten steps? Since three steps bring us back to where we started (because 1 is in a cycle of length three), nine steps take us around the cycle three times and the tenth takes us to 5. Thus $1 \to 5$ in the tenth power. Similarly, $5 \to 3$ and $3 \to 1$. Clearly $7 \to 7$ regardless of the power. Ten steps take us around the cycle $(2,6)$ exactly five times, so $2 \to 2$ and $6 \to 6$. Thus the tenth power is $(1,5,3)(7)(2)(6)$.

Suppose we have a permutation in cycle form whose cycle lengths all divide k. The reasoning in the previous paragraph shows that the kth power of that permutation will be the identity; that is, all the cycles will be 1-long and so every element is mapped to itself. In particular, if we are considering permutations of an n-set, every cycle has length at most n and so we can take $k = n!$ regardless of the permutation. We have shown

Theorem 2.1 *Given a set S, there is a k depending on $|S|$ such that f^k is the identity map for every permutation f of S.*

Without cycle notation, it would be harder to prove the theorem. $\square$

Example 2.8 Average cycle information What is the average number of elements fixed by a random permutation? More generally, what is the average number that belong to cycles of length k? What is the average number of cycles in a random permutation? We'll see that these questions are easy to answer.

Make the $n!$ permutations of $\{1, 2, \ldots, n\}$ into a probability space by giving each permutation probability $1/n!$—the uniform distribution. We want to define random variables X_i, one for each element of $\underline{n}$ such that their sum is the number of elements in k-cycles. Then $\mathbf{E}(X_1 + \cdots + X_n)$ is the average number of elements that belong to k-cycles. Let

$$X_i \;=\; \begin{cases} 1, & \text{if } i \text{ belongs to a } k\text{-cycle,} \\ 0, & \text{otherwise.} \end{cases}$$

Note that $\mathbf{E}(X_i)$ is the same for all i. By properties of expectation

$$\mathbf{E}(X_1 + \cdots + X_n) \;=\; \mathbf{E}(X_1) + \cdots + \mathbf{E}(X_n) \;=\; n\,\mathbf{E}(X_1).$$

It is shown in Exercise 2.2.2 that there are $(n-1)!$ permutations with $X_1 = 1$. Since each permutation has probability $1/n!$, $n\,\mathbf{E}(X_1) = n\frac{(n-1)!}{n!} = 1$. In other words, on average one element belongs to a k-cycle, regardless of the value of k as long as $1 \le k \le n$.

We now consider the average number of cycles in a permutation. To do this we want to define random variables, one for each element of $\underline{n}$, such that their sum is the number of cycles. Let

$$Y_i \;=\; \frac{1}{\text{length of the cycle containing } i}.$$

Since each element of a k-cycle contributes $1/k$ to the sum $Y_1 + \cdots + Y_n$, all k elements in the cycle contribute a total of 1. Thus $Y_1 + \cdots + Y_n$ is the number of cycles in the permutation. We have

$$\begin{aligned} \mathbf{E}(Y_1 + \cdots + Y_n) \;&=\; \mathbf{E}(Y_1) + \cdots + \mathbf{E}(Y_n) \;=\; n\,\mathbf{E}(Y_1) \\ &=\; n \sum_{k=1}^{n} \Pr(1 \text{ is in a } k\text{-cycle})\,\frac{1}{k} && \text{by definition of } \mathbf{E} \\ &=\; n \sum_{k=1}^{n} \frac{1}{n}\,\frac{1}{k} && \text{by Exercise 2.2.2} \\ &=\; \sum_{k=1}^{n} \frac{1}{k}. \end{aligned}$$

The last sum is approximately $\ln n$ because the sum is approximately $1 + \int_1^n \frac{dx}{x} \;=\; 1 + \ln n$. □

***Example 2.9 Involutions** An *involution* is a permutation which is equal to its inverse. Since $f(x) = f^{-1}(x)$, we have $f^2(x) = f(f^{-1}(x)) = x$. Thus involutions are those permutations which have all their cycles of lengths one and two. How many involutions are there on $\underline{n}$?

Let's count the involutions with exactly k 2-cycles and use the Rule of Sum to add up the results. We can build such an involution as follows:

- Select $2k$ elements for the 2-cycles AND
- partition these $2k$ elements into k blocks that are all of size 2 AND
- put the remaining $n - 2k$ elements into 1-cycles.

Since there is just one 2-cycle on two given elements, we can interpret each block as 2-cycle. This specifies f. The number of ways to carry out the first step is $\binom{n}{2k}$. The next step is trickier. A first guess might be simply the multinomial coefficient $\binom{2k}{2,\ldots,2} = (2k)!/2^k$. This leads to the dilemma of the poker hand with two pairs (Example 1.16): We're assuming an ordering on the pairs even though

they don't have one. For example, with $k = 3$ and the set $\underline{6}$, there are just 15 possible partitions as follows.

$$\begin{array}{lll}
\{\{1,2\},\{3,4\},\{5,6\}\} & \{\{1,2\},\{3,5\},\{4,6\}\} & \{\{1,2\},\{3,6\},\{4,5\}\} \\
\{\{1,3\},\{2,4\},\{5,6\}\} & \{\{1,3\},\{2,5\},\{4,6\}\} & \{\{1,3\},\{2,6\},\{4,5\}\} \\
\{\{1,4\},\{2,3\},\{5,6\}\} & \{\{1,4\},\{2,5\},\{3,6\}\} & \{\{1,4\},\{2,6\},\{3,5\}\} \\
\{\{1,5\},\{2,3\},\{4,6\}\} & \{\{1,5\},\{2,4\},\{3,6\}\} & \{\{1,5\},\{2,6\},\{3,4\}\} \\
\{\{1,6\},\{2,3\},\{4,5\}\} & \{\{1,6\},\{2,4\},\{3,5\}\} & \{\{1,6\},\{2,5\},\{3,4\}\}
\end{array}$$

This is smaller than $\binom{6}{2,2,2} = 6!/2!2!2! = 90$ because all 3! ways to order the three blocks in each partition are counted differently. This is because we've chosen a first, second and third block instead of simply dividing $\underline{6}$ into three blocks of size two.

How can we solve the dilemma? Actually, the discussion of what went wrong contains the key to the solution: The multinomial coefficient counts ordered collections of k blocks and we want unordered collections. Since the blocks in a partition are all distinct, there are $k!$ ways to order the blocks and so the multinomial coefficient counts each unordered collection $k!$ times. Thus we must simply divide the multinomial coefficient by $k!$.

If this dividing by $k!$ bothers you, try looking at it this way. Let $f(k)$ be the number of ways to carry out Step 2. Since the k blocks can be permuted in $k!$ ways, the Rule of Product tells us that there are $f(k)\,k!$ ways to select k ordered blocks of 2 elements each. Thus $f(k)\,k! = \binom{2k}{2,\ldots,2}$.

Since there is just one way to carry out Step 3, the Rule of Product tells us that the number of involutions is

$$\binom{n}{2k}\,\frac{1}{k!}\,\binom{2k}{2,\ldots,2} \;=\; \frac{n!}{(2k)!(n-2k)!}\,\frac{1}{k!}\,\frac{(2k)!}{(2!)^k}.$$

Simplifying and using the Rule of Sum to combine the various possible values of k, we obtain

Theorem 2.2 *The number of involutions of $\underline{n}$ is*

$$\sum_{k=0}^{\lfloor n/2\rfloor} \frac{n!}{(n-2k)!2^k k!}.$$

The notation $\lfloor x\rfloor$ stands for the *largest integer* in x; that is, the largest integer $m \le x$. It is also called the *floor* of x. □

***Example 2.10 Permutation matrices and parity** This example assumes some familiarity with matrices and determinants.

Suppose f and g are permutations of $\underline{n}$. We can define an $n \times n$ matrix F to consist of zeroes except that the (i,j)th entry, $F_{i,j}$, equals one whenever $f(j) = i$. Define G similarly. Then

$$(FG)_{i,j} \;=\; \sum_{k=1}^{n} F_{i,k}G_{k,j} \;=\; F_{i,g(j)},$$

since $G_{k,j} = 0$ except when $g(j) = k$. By the definition of F, this entry of F is zero unless $f(g(j)) = i$. Thus $(FG)_{i,j}$ is zero unless $(fg)(j) = i$, in which case it is one. We have proven that FG corresponds to fg. In other words:

Composition of permutations corresponds to multiplication of matrices.

It is also easy to prove that f^{-1} corresponds to F^{-1}. For example, the permutations $f = (1,3,2)(4)$ and $g = (1,2,3,4)$, written in cycle form, correspond to

$$F \;=\; \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad G \;=\; \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad \text{while} \quad FG \;=\; \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix},$$

which corresponds to the cycle form $(1)(2)(3,4)$, which equals fg.

Using this correspondence, we can prove things such as $(fg)^{-1} = g^{-1}f^{-1}$ and $(f^k)^{-1} = (f^{-1})^k$ by noting that they are true for matrices F and G.

Note that, since the matrix F contains exactly one 1 in each row and column, its determinant is either $+1$ or -1.

Definition 2.4 Parity of a permutation *If the matrix F corresponds to the permutation f, we call $\det F$ the* **parity** *of f and write it $\chi(f)$. If $\chi(f) = +1$, we say f is* **even** *and, if $\chi(f) = -1$, we say that f is* **odd**.

The rest of this example is devoted to proving:

Theorem 2.3 Properties of parity *Suppose f and g are permutations of $\underline{n}$. The following are true.*

(a) $\chi(fg) = \chi(f)\,\chi(g)$.

(b) *If g is f with the elements of $\underline{n}$ relabeled, then $\chi(f) = \chi(g)$.*

(c) *If f is written a product of (not necessarily disjoint) cycles, then $\chi(f) = (-1)^k$, where k is the number of cycles in the product that have even length. In particular, if f is written as a product of k 2-cycles, then $\chi(f) = (-1)^k$.*

(d) *For $n > 1$, exactly half the permutations of $\underline{n}$ are even and exactly half are odd.*

Let F and G be the matrices corresponding to f and g. Then (a) follows from $\det(FG) = \det(F)\,\det(G)$.

In the next paragraph, we'll show that a relabelling can be written $G = P^tFP$ where P is permutation matrix and P^t is the transpose of P. Take determinants:

$$\chi(g) \;=\; \det G \;=\; \det(P^tFP) \;=\; \det P^t\,\det F\,\det P.$$

Now $\det P^t = \det P$ for any matrix P and $(\det P)^2 = \det P$ since the determinant of a permutation matrix is $+1$ or -1. Thus we have $\chi(g) \;=\; \det F \;=\; \chi(f)$.

We must prove the relabeling claim. To relabel, we must permute the columns of F in some fashion and permute the rows of F in the same way. You should convince yourself that multiplying a matrix M by a permutation matrix P gives a matrix MP in which the columns of M have been permuted. Permuting the rows of F is the same as permuting the columns of F^t, except that the resulting matrix is transposed. Thus $(F^tP)^t$ permutes the rows of F and so the rows and columns are permuted by $(F^tP)^tP \;=\; P^tFP$.

If $f = c_1 \cdots c_k$, then $\chi(f) = \chi(c_1) \cdots \chi(c_k)$. Thus (c) follows if we know the parity of a single cycle c. With appropriate relabeling, the matrix corresponding to an m-cycle c is the $m \times m$ matrix

$$C_m \;=\; \begin{pmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \\ 1 & 0 & 0 & 0 & \cdots & 0 \end{pmatrix}$$

Expanding about the first row, we have $\det C_m = -\det C_{m-1}$. Since $\det C_2 = -1$, it follows that $\det C_m = (-1)^{m-1}$. Thus c_m is odd if and only if m is even. This completes the proof of (c).

Let $\mathcal{P}_o$ and $\mathcal{P}_e$ be the sets of odd and even permutations of $\underline{n}$, respectively. Since $n \geq 2$, the permutation $t = (1,2)$ is among the permutations. Since $\chi(t) = -1$, it follows from (c) that $T_o : f \mapsto tf$ is map from $\mathcal{P}_o$ to $\mathcal{P}_e$. Since t^2 is the identity function, $T_e : g \mapsto tg$ is the inverse of T_o. You should be able to see that this implies that T_o and T_e are bijections and so $\mathcal{P}_o = \mathcal{P}_e$, proving (d).

For those familiar with group theory, the set of permutations of $\underline{n}$ is called the symmetric group on n symbols and the subset of even permutations is called the alternating group on n symbols. □

Exercises

2.2.1. This exercise lets you check your understanding of cycle form. The answers are given at the end of this problem set. A permutation is given in one-line, two-line or cycle form. Convert it to the other two forms. Give its inverse in all three forms. (The answers are given at the end of this problem set.)

(a) (1,5,7,8) (2,3) (4) (6).

(b) $\left(\begin{smallmatrix}1&2&3&4&5&6&7&8\\8&3&7&2&6&4&5&1\end{smallmatrix}\right)$.

(c) (5,4,3,2,1), which is in one-line form.

(d) (5,4,3,2,1), which is in cycle form.

2.2.2. Let f be a permutation of $\underline{n}$. The cycle of f that contains 1 is called the *cycle generated by* 1.

(a) Prove that the number of permutations in which the cycle generated by 1 has length n is $(n-1)!$.

(b) For $1 \le k \le n$, prove that the number of permutations in which the cycle generated by 1 has length k is $(n-1)!$, independent of the value of k. (Remember that a permutation must permute *all* of $\underline{n}$.)

(c) Conclude that if $1 \le k \le n$ and a permutation of $\underline{n}$ is selected uniformly at random, then the probability that 1 belongs to a k-cycle is $1/n$, independent of k.

2.2.3. A carnival barker has four cups upside down in a row in front of him. He places a pea under the cup in the first position. He quickly interchanges the cups in the first and third positions, then the cups in the first and fourth positions and then the cups in the second and third positions. This entire set of interchanges is done a total of five times. Where is the pea?
Hint. Write one entire set of interchanges as a permutation in cycle form.

2.2.4. Let $P_k(n)$ be the number of permutation of $\underline{n}$ all of whose cycles have length at most k. Thus $P_1(n) = 1$ since only the identity permutation has all cycles of length 1. Also, $P_2(n)$ is the number of involutions. For later convenience, define $P_k(0) = 1$.

(a) By considering the cycle containing $n+1$, prove that $P_2(n+1) = P_2(n) + nP_2(n-1)$ for $n \ge 0$

(b) State and prove a similar recursion for P_3.

2.2.5. A *fixed point* of a permutation f is an element x of the domain such that $f(x) = x$. A *derangement* is a permutation f with no fixed points; i.e., $f(x) \ne x$ for all x.

(a) Prove that the probability that a random permutation f of $\underline{n}$ has $f(k) = k$ equals $1/n$.

(b) If we treat the n events $f(1) \ne 1, \ldots, f(n) \ne n$ as independent, what is the probability that f is a derangement? Conclude that we might expect approximately $n!/e$ derangements of $\underline{n}$. In Example 4.5 (p. 99), you'll see that this heuristic estimate is extremely accurate.

(c) Let D_n be the number of derangements of $\underline{n}$. Prove that the number of permutations of $\underline{n}$ with exactly k fixed points is $D_{n-k}\binom{n}{k} \approx \frac{n!}{k!\,e}$.

2.2.6. Let $z(n,k)$ be the number of permutations of $\underline{n}$ having exactly k cycles. These are called the *signless Stirling numbers of the first kind*. Our notation is not standard. The notation $s(n,k)$ is commonly used both for these and for the Stirling numbers of the first kind, which may differ from them in sign.

(a) Prove the recursion

$$z(n+1,k) = \sum_{i=0}^{n} \binom{n}{i} i!\; z(n-i,k-1).$$

(b) Give initial conditions and the range of n and k for which the recursion is valid.

(c) Construct a table of $z(n,k)$ for $n \le 5$. Note: You can obtain a partial check on your calculations by using $\sum_{k\ge 0} z(n,k) = n!$.

2.2.7. We want to compute the average of $|a_1 - a_2| + |a_2 - a_3| + \cdots + |a_{n-1} - a_n|$ over all permutations $(a_1, \ldots, a_n)$ of $\underline{n}$.

(a) Show that the average equals

$$\frac{2}{n} \sum_{i \le i \le j \le n} (j - i).$$

(b) Show that the answer is $\frac{n^2-1}{2}$.

Answers

2.2.1. (a) $\left(\begin{smallmatrix} 1&2&3&4&5&6&7&8 \\ 5&3&2&4&7&6&8&1 \end{smallmatrix}\right)$ is the two line form and (5,3,2,4,7,6,8,1) is the one-line form. (We'll omit the two-line form in the future since it is simply the one line form with $1, 2, \ldots$ placed above it.) The inverse is (1,8,7,5) (2,3) (4) (6) in cycle form and (8,3,2,4,1,6,5,7) in one-line form.

(b) The cycle form is (1,8) (2,3,7,5,6,4). The inverse in cycle form is (1,8) (2,4,6,5,7,3) and in one-line form is (8,4,2,6,7,5,3,1).

(c) The cycle form is (1,5) (2,4) (3). The permutation is its own inverse.

(d) This is not the standard form for cycle form. Standard form is (1,5,4,3,2). The one-line form is (5,1,2,3,4). The inverse is (1,2,3,4,5) in cycle form and (2,3,4,5,1) in one line form.

2.3 Other Combinatorial Aspects of Functions

Monotonic Functions and Unordered Lists

The one-line notation for a function is simply an (ordered) list, so there is a simple correspondence (i.e., bijection) between lists and functions: A k-list from S is a function $f: \underline{k} \to S$. If f is an injection, the list has no repeats.

How can we get unordered lists to correspond to functions? (Recall that unordered lists correspond to sets or multisets depending on whether repeats are forbidden or not.) The secret is a two step process. First, think of a unique way to order the list, say $s_1, s_2, \ldots, s_k$. Second, interpret the resulting list as a one-line function as done above. In mathematics, people refer to a unique thing (or process or whatever) that has been selected as *canonical*. Thus one would probably speak of a canonical ordering of the list rather than a unique ordering; however, both terms are correct.

Let's look at a small example. Here's a listing of all $\binom{5+3-1}{3} = 35$ of the unordered 3-lists with repetition from $\underline{5}$. In the listing, an entry like 2,5,5 stands for the 3-list containing one 2 and two 5's.

1,1,1	1,1,2	1,1,3	1,1,4	1,1,5	1,2,2	1,2,3	1,2,4	1,2,5	1,3,3	1,3,4	1,3,5
1,4,4	1,4,5	1,5,5	2,2,2	2,2,3	2,2,4	2,2,5	2,3,3	2,3,4	2,3,5	2,4,4	2,4,5
2,5,5	3,3,3	3,3,4	3,3,5	3,4,4	3,4,5	3,5,5	4,4,4	4,4,5	4,5,5	5,5,5	

We've simply arranged the elements in each of the 3-lists to be in "nondecreasing order." Let $b_1, b_2, \ldots, b_k$ be an ordered list. We say the list is in *nondecreasing order* if the values are not decreasing as we move from one element to the next; that is, if $b_1 \le b_2 \le \cdots \le b_k$. We say that $f \in \underline{n}^{\underline{k}}$ is a *nondecreasing function* if its one-line form is nondecreasing; that is, $f(1) \le f(2) \le \cdots \le f(k)$. The list of lists we've created is a bijection between (i) the unordered 3-lists with repetition from $\underline{5}$ and (ii) the nondecreasing functions in $\underline{5}^{\underline{3}}$ written in one-line notation. Thus, 3-multisets of $\underline{5}$ correspond to nondecreasing functions.

In a similar fashion we say that

$$\text{the list } b_1, b_2, \ldots, b_k \text{ is in } \left\{ \begin{array}{c} \textit{nonincreasing} \\ \textit{decreasing} \\ \textit{increasing} \end{array} \right\} \textit{ order if } \left\{ \begin{array}{l} b_1 \ge b_2 \ge \cdots \ge b_k; \\ b_1 > b_2 > \cdots > b_k; \\ b_1 < b_2 < \cdots < b_k. \end{array} \right.$$

Again, this leads to similar terminology for functions. All such functions are also called *monotone functions*. Some people say "strictly decreasing" when we say "decreasing," and likewise for (strictly) increasing. This is a good practice because it helps avoid confusion, therefore we'll usually do it. Also, people say *weakly decreasing* instead of nonincreasing, a convention we often adopt.

We can interchange the strings "decreas" and "increas" in the previous paragraphs and read the functions in the list backwards. Do it. In summary:

$$\left.\begin{array}{rcl} \text{nonincreasing} & = & \text{weakly decreasing} \\ \text{decreasing} & = & \text{strictly decreasing} \\ \text{nondecreasing} & = & \text{weakly increasing} \\ \text{increasing} & = & \text{strictly increasing} \end{array}\right\} \text{ means } \left\{\begin{array}{l} b_1 \geq b_2 \geq \cdots \geq b_k; \\ b_1 > b_2 > \cdots > b_k; \\ b_1 \leq b_2 \leq \cdots \leq b_k; \\ b_1 < b_2 < \cdots < b_k. \end{array}\right.$$

In the bijection we gave from 3-lists to functions, the 3-lists without repetition correspond to the strictly increasing functions. For example, 1,2,3 and 1,3,4 correspond to strictly increasing functions written in one-line notation. Thus 3-subsets of $\underline{5}$ correspond to strictly increasing functions. The bijection between 3-subsets of $\underline{5}$ and strictly decreasing functions is given by

$$3,2,1 \quad 4,2,1 \quad 4,3,1 \quad 4,3,2 \quad 5,2,1 \quad 5,3,1 \quad 5,3,2 \quad 5,4,1 \quad 5,4,2 \quad 5,4,3.$$

The function $(4,3,1)$ corresponds to the 3-subset $\{4,3,1\} = \{1,3,4\}$ of $\underline{5}$.

All these things are special cases of

Theorem 2.4 *There is a bijection between unordered k-lists with repetition made from $\underline{n}$ and the weakly increasing (resp. weakly decreasing) functions in $\underline{n}^{\underline{k}}$. In this correspondence, lists without repetition (i.e., sets) correspond to the strictly increasing (resp. strictly decreasing) functions.*

Example 2.11 A bijection between strict and nonstrict functions Let $m = n+k-1$. We will construct a bijection between the weakly decreasing functions in $\underline{n}^{\underline{k}}$ and the strictly decreasing functions in $\underline{m}^{\underline{k}}$.

Let $f \in \underline{n}^{\underline{k}}$ be a weakly decreasing function. Define a function g by $g(i) = f(i) + k - i$. This is a strictly decreasing function because

$$g(i) - g(i+1) \;=\; \Big(f(i) + k - i\Big) - \Big(f(i+1) + k - (i+1)\Big) \;=\; 1 + \Big(f(i) - f(i+1)\Big) \;\geq\; 1.$$

It has the same domain, $\underline{k}$, as f, but its codomain is $\underline{n+k-1} = \underline{m}$ because $g(1)$ is $k-1$ larger than $f(1)$ and $f(1)$ may be as large as n. Let's give this map from weakly decreasing functions in $\underline{n}^{\underline{k}}$ to strictly decreasing functions in $\underline{n-k+1}^{\underline{k}}$ the name φ. We will prove that φ is a bijection.

It is easy to see that $f_1 \neq f_2$ implies that $\varphi(f_1) \neq \varphi(f_2)$ and so φ is an injection. We claim it is a bijection. To see this, suppose that $h \in \underline{m}^{\underline{k}}$ is strictly decreasing. Define f by $f(i) = h(i) - (k-i)$. This is a function in $\underline{n}^{\underline{k}}$ with $\varphi(f) = h$. To complete the proof, we must prove that it is weakly decreasing. We have

$$f(i) - f(i+1) \;=\; \Big(h(i) - h(i+1)\Big) - 1 \;\geq\; 1 - 1 \;=\; 0.$$

We can combine our knowledge that φ is a bijection with the Theorem 2.4 to obtain another proof for the formula for the number of k-multisets that can be formed from $\underline{n}$. By Theorem 2.4, the number of k-multisets that can be formed from $\underline{n}$ is the same as the number of weakly decreasing functions from $\underline{k}$ to $\underline{n}$. By φ, the latter equals the number of strictly decreasing functions from $\underline{k}$ to $\underline{m}$. By the theorem, this equals the number of k-subsets of $\underline{m}$. Since this last number is $\binom{m}{k} = \binom{n+k-1}{k}$, there are that many k-multisets that can be formed from $\underline{n}$. This is very similar to a proof that was given after Theorem 1.8 (p. 37). □

Example 2.12 Unimodal sequences A sequence that is first weakly increasing and then weakly decreasing is sometimes called unimodal (a term which is not strictly correct). The weakly increasing or weakly decreasing part may be empty. Here are some examples of unimodal sequences:

$$1,3,3,4,2 \qquad 1,1,1,1 \qquad 1,3,5,5,4,2,2 \qquad 1,2,3,2,1.$$

A variety of counting questions can be asked about unimodal sequences. Generally they can all be handled by the following method:

1. Imagine breaking the sequence into three pieces:

$$\underbrace{a_1 \le a_2 \le \cdots \le a_{k-1}}_{k-1 \text{ items}} \quad < \quad a_k \quad \ge \quad \underbrace{a_{k+1} \ge a_{k+2} \cdots \ge a_\ell}_{\ell-k \text{ items}},$$

   where a_k is the first occurrence of the largest element in the sequence.

2. Obtain a formula for the number of such sequences based on the value of a_k and, possibly, on the values of k and ℓ.

3. Sum the result of the previous step.

We'll do a couple of examples.

How many monotonic sequences are there in which the inequalities are all strict and the elements lie in $\underline{n}$? Suppose the largest element is t. The sequence elements preceding t must be a strict monotonic sequence with all elements in $\underline{t-1}$. Since such sequences correspond to subsets of $\underline{t-1}$ there are 2^{t-1} choices for such a sequence. Similarly, there are 2^{t-1} choices for the elements after t. Thus there are $(2^{t-1})^2$ strict monotonic sequences of positive integers with largest element t. Hence the answer to the original question is

$$\sum_{t=1}^{n}(2^{t-1})^2 \;=\; 1+4+16+\cdots+4^{n-1} \;=\; \frac{1-4^n}{1-4} \;=\; \frac{4^n-1}{3}.$$

How many weakly monotonic sequences of length ℓ are there whose elements lie in $\underline{n}$? Let $a_k = t$.

- The weakly increasing sequence preceding a_k corresponds to a $(k-1)$-multiset formed from $\underline{t-1}$. There are $\binom{t+k-3}{k-1}$ of them. We have to be careful here because of the case $t = 1$.
- It turns out to be easier to treat the case $t = 1$ separately. There is only one weakly monotonic sequence of length ℓ with largest term 1, namely the sequence of all ones.
- The weakly decreasing sequence following a_k corresponds to a $(\ell-k)$-multiset formed from $\underline{t}$. There are $\binom{\ell+t-k-1}{\ell-k}$ of them.

Thus the number of weakly monotonic sequences of length ℓ whose elements lie in $\underline{n}$ is

$$1+\sum_{t=2}^{n}\sum_{k=1}^{\ell}\binom{t+k-3}{k-1}\binom{\ell+t-k-1}{\ell-k}. \;\square$$

Image and Coimage

Again, let $f: A \to B$ be a function. The *image* of f is the set of values f actually takes on: $\text{Image}(f) = \{\, f(a) \mid a \in A \,\}$. The definition of a surjection can be rewritten $\text{Image}(f) = B$.*

For each $b \in B$, the *inverse image* of b, written $f^{-1}(b)$ is the set of those elements in A whose image is b; i.e.,

$$f^{-1}(b) \;=\; \{\, a \mid a \in A \text{ and } f(a) = b \,\}.$$

This extends our earlier definition of f^{-1} from bijections to all functions; however, such an f^{-1} can't be thought of as a function from B to A unless f is a bijection because it will not give a unique $a \in A$ for each $b \in B$. (There is a slight abuse of notation here: If $f: A \to B$ is a bijection, our new notation is $f^{-1}(b) = \{a\}$ and our old notation is $f^{-1}(b) = a$.)

The collection of nonempty inverse images of elements of B is called the *coimage* of f.

We claim that the coimage of f is the partition of A whose blocks are the maximal subsets of A on which f is constant. For example, if $f \in \{a, b, c\}^{\underline{5}}$ is given in one-line form as (a, c, a, a, c), then

$$\text{Coimage}(f) \;=\; \{f^{-1}(a), f^{-1}(c)\} \;=\; \big\{\{1, 3, 4\}, \{2, 5\}\big\},$$

f is a on $\{1, 3, 4\}$ and is c on $\{2, 5\}$.

We now prove the claim. If $x \in A$, let $y = f(x)$. Then $x \in f^{-1}(y)$ and so the union of the nonempty inverse images contains A. Clearly it does not contain anything which is not in A. If $y_1 \neq y_2$, then we cannot have $x \in f^{-1}(y_1)$ and $x \in f^{-1}(y_2)$ because this would imply $f(x) = y_1$ and $f(x) = y_2$, a contradiction. Thus $\text{Coimage}(f)$ is a partition of A. Clearly x_1 and x_2 belong to the same block if and only if $f(x_1) = f(x_2)$. Hence a block is a maximal set on which f is constant.

Example 2.13 Blocks and Stirling numbers How many functions in B^A have a coimage with exactly k blocks?

This means that the coimage is a partition of A having exactly k blocks. Recall that $S(n, k)$, a Stirling number of the second kind, denotes the number of partitions of an n-set into k blocks. (See Example 1.27 (p. 34).) There are $S(|A|, k)$ ways to choose the blocks of the coimage. The partition of A does not fully specify a function $f \in B^A$. To complete the specification, we must specify the image of the elements in each block, in other words, an injection from the set of blocks to B. This is an ordered selection of size k without replacement from B. There are $|B|!/(|B| - k)!$ such injections, independent of which k block partition of A we are considering. By the Rule of Product, there are $S(|A|, k)\big(|B|!/(|B| - k)!\big)$ functions $f \in B^A$ with $|\,\text{Coimage}(f)| = k$. □

We can describe the image and coimage of a function by the arrow pictures introduced at the end of Example 2.4. $\text{Image}(f)$ is the set of those $b \in B$ which appear as labels of arrowheads. A block in $\text{Coimage}(f)$ is the set of labels on the tails of those arrows that all have their heads pointing to the same value; for example, the block of $\text{Coimage}(f)$ arising from $b \in \text{Image}(f)$ is the set of labels on the tails of those arrows pointing to b.

* The image is also called the range. Unfortunately, the codomain is also sometimes called the range.

Example 2.14 Blocks of given sizes Let $\vec{b}$ be a vector of nonnegative integers such that $\sum_{k\geq 1} kb_k = n$, and let $B(n,\vec{b})$ be the number of partitions of $\underline{n}$ having exactly b_k blocks of size k. We call $\vec{b}$ the *type* of a partition and so $B(n,\vec{b})$ is the number of partitions of type $\vec{b}$.

Consider the possible coimages of functions $f\colon \underline{n} \to \underline{m}$. Since the coimage is a partition of $\underline{n}$, we can talk about the type of the coimage, too. There is a restriction: Since the coimage can't have more blocks than the size of the codomain, $b_1 + 2b_2 + \cdots \leq m$.

We want to compute $B(n,\vec{b})$. To do this, we first partition $\underline{n}$ into ordered blocks where there are kb_k elements in block k for each k. Next, for each k, we partition block k into b_k blocks each of size k. For the first step, all we need is a multinomial coefficient since that step is precisely the multinomial coefficient setup. The second step would also be a multinomial coefficient setup if we had put the blocks of size k in an ordered list. Since there are $b_k!$ ways to order the b_k blocks of size k, we need to divide that multinomial coefficient by $b_k!$. This gives us

$$B(n,\vec{b}) \;=\; \binom{n}{b_1, 2b_2, 3b_3, \ldots} \prod_{b_k\neq 0} \frac{1}{b_k!} \binom{kb_k}{\underbrace{k,k,\ldots,k}_{b_k \text{ copies of } k}} \;=\; \frac{n!}{\prod_{b_k\neq 0} b_k!\,(k!)^{b_k}}. \qquad \square$$

The Pigeonhole Principle

The *Pigeonhole Principle* is a method for obtaining statements of the form

$$\text{If } n \geq g(d), \;\; \text{then } \mathcal{A}(n,d),$$

where $\mathcal{A}$ is a statement and g is some function depending on $\mathcal{A}$. For example, we'll see that, if $n \geq d^2+1$, then any sequence of n distinct numbers contains a monotonic subsequence of length d. (What follows "then" is $\mathcal{A}(n,d)$.)

Here is a statement of the principle in two forms.

Theorem 2.5 Pigeonhole Principle Function form: *Suppose A and B are sets with $|A| > |B|$, then for every function $f : A \to B$ there is a $b \in B$ with $|f^{-1}(b)| > 1$.*
Partition form: *Suppose $\mathcal{P}$ is a partition of the set A into less than $|A|$ blocks. Then some block contains more than one element of A.*

You should be able to prove this theorem. In fact, it is so simple we should probably not even call it a theorem. To see why the two forms of the theorem are equivalent, first suppose $f : A \to B$. Let $\mathcal{P}$ be the coimage of f. It must have at most $|B| < |A|$ blocks. Conversely suppose $\mathcal{P}$ is a partition of A. Number the blocks in some fashion from 1 to $|\mathcal{P}|$. Let $B = \{1,\ldots,|\mathcal{P}|\}$ and define $f : A \to B$ by letting $f(a)$ be the number of the block that contains A.

Where did the rather strange name "Pigeonhole Principle" come from? Old style desks often had what looked like a stacked array of boxes that were open in the front. These boxes were usually used to hold various letters and folded or rolled papers. The boxes were called pigeon holes because of their resemblance to nesting boxes in pigeon coops. If $|A|$ letters are placed in $|B|$ pigeonholes in a desk and $|A| > |B|$, then at least one pigeonhole contains at least two documents.

Example 2.15 Subset sums Given a set P of integers, let $\sum P$ be the sum of the elements of P. We say that a set S of positive integers has the *two-sum property* if there are subsets P and Q of S with $P \neq Q$ and $\sum P = \sum Q$. Not all sets have the two-sum property. For example, $\{1, 2, 4, 8\}$ does not, but $\{1, 2, 4, 5\}$ has the two-sum property—take $P = \{1, 4\}$ and $Q = \{5\}$. The set $S = \{1, 2, 4, 8\}$ fails because its elements grow too rapidly. Can we put some condition on $|S|$ and $\max S$, the largest element of S, so that S must have the two sum property? Since n can't be too large, we might expect a statement of the form

$$\text{If } |S| \geq k \text{ and } \max S \leq h(k), \text{ then } S \text{ has the two-sum property.} \qquad 2.1$$

If a subset of S has the two-sum property, then so does S. Hence it suffices to prove (2.1) when $|S| = k$ since any S with $|S| > k$ can replaced by a subset S' of size k. Thus (2.1) is equivalent to

$$\text{If } |S| = k \text{ and } \max S \leq h(k), \text{ then } S \text{ has the two-sum property.}$$

Since we want to look for repeated values of subset sums, our function f will map subsets to their sums. Thus $A = 2^S$, the subsets of S, and $f : A \to \{0, 1, \ldots, \sum S\}$ is defined by $f(P) = \sum P$.

To apply the pigeonhole principle we need the cardinality of the domain and codomain of f. We have $|A| = 2^{|S|} = 2^k$. Since $B = \{0, 1, \ldots, \sum S\}$, we have $|B| = \sum S + 1$. For the Pigeonhole Principle, we need $2^k > \sum S + 1$. To get this, we need an upper bound for $\sum S$. Let $\max S = n$ and notice that

$$\sum S \;\leq\; \underbrace{n + (n-1) + (n-2) + \cdots + (n-k+1)}_{k \text{ terms}} \;\leq\; nk.$$

Thus it suffices to have $2^k \geq nk + 1$. This is easily solved to get an inequality for n. Recalling that $n = \max S$ and $k = |S|$, we have proved that any set S of positive integers has the two-sum property if

$$\max S \;\leq\; \frac{2^{|S|} - 1}{|S|}. \quad \square$$

Example 2.16 Monotonic subsequences We will prove

Theorem 2.6 *Given a sequence of more than mn distinct numbers, there must be either a subsequence of $n + 1$ decreasing numbers or a subsequence of $m + 1$ increasing numbers.*

A subsequence need not be consecutive; for example, 1,2,3 is a subsequence of 7,1,6,4,2,4,3.

The theorem is best possible because the following sequence contains nm numbers, the longest decreasing subsequence has length n and the longest increasing subsequence has length m.

$$n, (n-1), \ldots, 1, \quad 2n, (2n-1), \ldots, n+1, \quad 3n, (3n-1), \ldots, 2n+1, \quad \ldots, \quad mn, (mn-1), \ldots, (m-1)n+1. \qquad 2.2$$

How can we prove the theorem? Here is a fairly natural approach which turns out to be incorrect. We could try to "grow" sequences as follows. Let $a_1, \ldots a_\ell$ be the given sequence. Then a_ℓ is both an increasing and a decreasing sequence starting at ℓ. We back up from ℓ one step at a time until we reach 1. Suppose we have decreasing and increasing sequences starting at t. If $a_{t-1} < a_t$, we can put a_{t-1} at the front of our increasing sequence and increase its length by one. If $a_{t-1} > a_t$ we can put a_{t-1} at the front of our decreasing sequence and increase its length by one. Each step increases one length or the other by 1. Thus, when we reach a_1, the sum of the lengths is the initial sum plus the number of steps: $2 + (\ell - 1) = \ell + 1$. Thus, if $\ell \geq m + n$, we have either an increasing subsequence longer than m or a decreasing one longer than n. This can't be right—it's better than the claim in the theorem and (2.2) tells us that's best possible. Can you see what is wrong?

* * * Stop and think about this! * * *

We must compare a_{t-1} with the starts of the increasing and decreasing subsequences that we've built so far and, when $t < \ell$ only one of these subsequences begins with a_t.

Can we salvage anything from what we did? Suppose no decreasing subsequence is longer than n and no increasing subsequence is longer than m. Let I_t be the length of the longest increasing subsequence that starts with a_t and let D_t be the length of the longest decreasing subsequence that starts with a_t. Define a map $f : \underline{\ell} \to \underline{m} \times \underline{n}$ by $f(t) = (I_t, D_t)$. We'll show that f is an injection. By the Pigeonhole Principle, we cannot have $|A| > |B|$. In other words $\ell \leq m \times n$. The contrapositive is the theorem because it says that if $\ell > m \times n$, then it is not true that both (a) no increasing subsequence exceeds m and (b) no decreasing subsequence exceeds n. In other words, if $\ell > m \times n$, there is either an increasing subsequence longer than m or a decreasing subsequence longer than n.

It remains to show that f is an injection. Suppose $i < j$. You should be able to see that

$$\text{If } a_i < a_j, \text{ we have } I_i > I_j.$$
$$\text{If } a_i > a_j, \text{ we have } D_i > D_j.$$

This completes the proof. Another proof is given in the exercises. □

Exercises

2.3.1. This exercise lets you check your understanding of the definitions. In each case below, some information about a function is given to you. Answer the following questions and give reasons for your answers: (The answers are given at the end of this problem set.)

(i) Have you been given enough information to specify the function; i.e., would this be enough data for a function envelope?

(ii) Can you tell whether or not the function is an injection? a surjection? a bijection? If so, what is it?

(a) $f \in \underline{4}^{\underline{5}}$, $\text{Coimage}(f) = \{\{1,3,5\},\{2,4\}\}$.

(b) $f \in \underline{5}^{\underline{5}}$, $\text{Coimage}(f) = \{\{1\},\{2\},\{3\},\{4\},\{5\}\}$.

(c) $f \in \underline{4}^{\underline{5}}$, $f^{-1}(2) = \{1,3,5\}$, $f^{-1}(4) = \{2,4\}\}$.

(d) $f \in \underline{4}^{\underline{5}}$, $\big|\text{Image}(f)\big| = 4$.

(e) $f \in \underline{4}^{\underline{5}}$, $\big|\text{Image}(f)\big| = 5$.

(f) $f \in \underline{4}^{\underline{5}}$, $\big|\text{Coimage}(f)\big| = 5$.

2.3.2. Let A and B be finite sets and let $f\colon A \to B$ be a function. Prove the following claims.

(a) $|\,\text{Image}(f)| = |\,\text{Coimage}(f)|$.

(b) f is an injection if and only if $|\,\text{Image}(f)| = |A|$.

(c) f is a surjection if and only if $|\,\text{Coimage}(f)| = |B|$.

2.3.3. Let $\vec{b}$ be a vector of nonnegative integers such that $\sum_{k\geq 1} kb_k = n$, let $C(n,\vec{b})$ be the number of permutations of $\underline{n}$ having exactly b_k cycles of length k, and let $B(n,\vec{b})$ be the number of partitions of $\underline{n}$ having exactly b_k blocks of size k. Prove that

$$C(n,\vec{b}) \;=\; B(n,\vec{b}) \prod_{k\geq 1} (k-1)!^{b_k}.$$

2.3.4. How many strictly unimodal sequences are there in which the largest element is in the exact middle of the sequence and no element exceeds n?

2.3.5. How can we get a bijection between partitions of a set and a reasonable class of functions? Does the notion of coimage help? What about partitions with exactly k blocks? This exercise deals with these issues.

(a) Given a partition of $\underline{n}$, let block 1 be the block containing 1. If the first k blocks have been defined, let block $k+1$ be the remaining block that contains the smallest remaining element. This numbers the blocks of a set partition. Number the blocks of the partition

$$\{3,9,2\},\ \{4,1,6\}\ \{5\}\ \{7,8\}.$$

(b) Given a partition B of a $\underline{n}$, associate with it a function $f \in \underline{n}^{\underline{n}}$ as follows. Number the blocks of B as above. Let $f(i)$ be the number of the block of B that contains i. Prove that if two partitions are different, then the functions associated with them are different.

(c) What is the coimage of the function associated with a partition B of $\underline{n}$? (Phrase your answer as simply as possible in terms of B.)

(d) Call a function $f \in \underline{n}^{\underline{n}}$ a *restricted growth function* if $f(1) = 1$ and $f(i) - 1$ is at most the maximum of $f(k)$ over all $k < i$. Which of the following functions in one-line form are restricted growth functions? Give reasons for your answers.

$$2,2,3,3 \quad 1,2,3,3,2,1 \quad 1,1,1,3,3 \quad 1,2,3,1.$$

(e) Prove that a function is associated with a partition of $\underline{n}$ if and only if it is a restricted growth function. (Since you already proved that different partitions have different functions associated with them, this is the desired bijection.)

(f) For $\underline{4}$, list in lexicographic order all restricted growth functions and, for each function, give the partition of $\underline{4}$ that corresponds to it.

2.3.6. Prove the generalized Pigeonhole Principle: Suppose that a set S is partitioned into k blocks. Prove that some block must have more than $(|S|-1)/k$ elements.

2.3.7. Let $f : A \to B$. Use the generalized Pigeonhole Principle to obtain a lower bound on the size of the largest block in the coimage of f.

2.3.8. Suppose $n+1$ numbers are selected from $\underline{2n}$. Prove that we must have selected two numbers such that one is a multiple of the other.
Hint. Write each number in the form $2^k m$ where m is odd. Study occurrences of values of m using the Pigeonhole Principle.

2.3.9. Prove Theorem 2.6 using the generalized Pigeonhole Principle (Exercise 2.3.6) as follows. Given $a_1, \ldots, a_\ell$, define $f : \underline{\ell} \to B$ by letting $f(t)$ be the length of the longest decreasing subsequence starting with a_t. If $i < j < \cdots$ and $f(i) = f(j) = \cdots$, look at the subsequence $a_i, a_j, \ldots$.

2.3.10. Given N, how large must t be so that every set S containing at least t positive integers satisfies the following condition? There are elements $a, b, c, d \in S$ such that (i) $a+b$ and $c+d$ have the same remainder when divided by N and (ii) $\{a,b\} \neq \{c,d\}$.
Hint. Look at remainders when sums of pairs are divided by N.

*2.3.11. Given a set S of integers, prove that the elements of some nonempty subset of S sum to a multiple of $|S|$.

Answers

2.3.1. (a) The domain and codomain of f are specified and f takes on exactly two distinct values. f is not an injection. Since we don't know the values f takes, f is not completely specified; however, it cannot be a surjection because it would have to take on all four values in its codomain.

(b) Since each block in the coimage has just one element, f is an injection. Since $|\operatorname{Coimage}(f)| = 5 = |\text{codomain of } f|$, f is a surjection. Thus f is a bijection and, since the codomain and domain are the same, f is a permutation. In spite of all this, we don't know the function; for example, we don't know $f(1)$, but only that it differs from all other values of f.

(c) We know the domain and codomain of f. From $f^{-1}(2)$ and $f^{-1}(4)$, we can determine the values f takes on the union $f^{-1}(2) \cup f^{-1}(4) = \underline{5}$. Thus we know f completely. It is neither a surjection nor an injection.

(d) This function is a surjection, cannot be an injection and has no values specified.

(e) This specification is nonsense. Since the image is a subset of the codomain, it cannot have more than four elements.

(f) This specification is nonsense. The number of blocks in the coimage of f equals the number of elements in the image of f, which cannot exceed four.

*2.4 Boolean Functions

A *Boolean function* f is a map from $\{0,1\}^n$ to $\{0,1\}$. Thus the domain of f is all n long vectors of zeroes and ones. Boolean functions arise in logic, where 0 is often replaced by F for "False" and 1 by T for "True." Boolean functions also arise in arithmetic, where 0 and 1 are digits of numbers in binary representation. Mathematically, there is no difference between these interpretations; however, the two different interpretations have slightly different notation associated with them.

Example 2.17 Basic Boolean functions Here are three functions from $\{0,1\}^2$ to $\{0,1\}$ in two-line form

$$p = \begin{pmatrix} (0,0) & (0,1) & (1,0) & (1,1) \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad s = \begin{pmatrix} (0,0) & (0,1) & (1,0) & (1,1) \\ 0 & 1 & 1 & 1 \end{pmatrix} \qquad f = \begin{pmatrix} (0,0) & (0,1) & (1,0) & (1,1) \\ 0 & 1 & 1 & 0 \end{pmatrix}.$$

If we think of x and y as integers, we can write $p(x,y) = xy$ and, indeed, this is the notation that is commonly used for p. To emphasize the multiplication, we might write $p(x,y) = x \cdot y$. Suppose that X and Y are statements; e.g., X may be the statement "It is cloudy." and Y may be "It is hot." We can build a more complicated statement from these two in many simple ways. One possibility is

"It is cloudy and it is hot."

We could abbreviate this to "X and Y." Let this compound statement be Z. Let $x = 0$ if X is false and $x = 1$ if X is true. Define y and z similarly. (This is the True/False interpretation of 0 and 1 mentioned earlier.) You should be able to see that $z = p(x,y)$ because Z is true if and only if both X and Y are true. Not surprisingly, the function p is called *and* in logic. Logicians sometimes write $p(x,y) = x \wedge y$ instead of $p(x,y) = xy$.

What interpretation does the function $s(x,y)$ have? If we write it as an arithmetic function on integers, it is a bit of a mess, namely, $x + y - xy$. In logic it has a very simple interpretation. Using the notation of the previous paragraph, let Z be the statement "X or Y." Our usual understanding of this is that Z is true if at least one of X and Y is true. This understanding is equivalent to the mathematical statement $s(x,y) = z$ since s is 1 if at least one of its arguments is 1. Logicians call this function *or*. There are two common ways to denote it: $s(x,y) = x \vee y$ and $s(x,y) = x + y$. The $x + y$ notation is a bit unfortunate because *it does not mean that x and y are to be added as integers*. Since the plus sign notation is commonly used in discussion of circuits, we will use it here. Remember:

If $x, y \in \{0,1\}$, then $x + y$ means OR, *not* addition.

How can our third function f be interpreted? In terms of logic, $f(x, y)$ corresponds to a statement which is true when exactly one of X and Y is true. Thus f is called the *exclusive or* and we write it as $f(x, y) = x \oplus y$. This function does not appear often in logic but it is important in binary arithmetic: $x \oplus y$ is the unit's (rightmost) digit of the binary sum of the numbers x and y. (Recall that the binary sum of 1 and 1 is 10.)

We need a little more notation. Negation, $n(x)$, also called complementation, is an important function. It is $n(x) = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$; i.e., $n(x)$ is true if and only if x is false. In terms of the previous notation, we can write $n(x) = x \oplus 1$. The notations x' and $\overline{x}$ are both commonly used for $n(x)$. We will use x'. You should be able to see quickly that $x'' = x$. □

Just as in ordinary algebra, we can combine the functions we introduced in the previous example. Some formulas such as $xy = yx$ and $(x + y)z = xz + yz$, which are true in ordinary arithmetic are true here, too. There are also some surprises such as $xx = x + x = x$ and $x + xy = x$. For those who like terminology, $xx = x$ and $x + x = x$ are called the idempotent laws for multiplication and addition and $x + xy = x$ is called the absorbtion law for addition. How can we check to see if an identity for Boolean expressions is correct? We can do this in one of two ways:

(a) (Brute force) By substituting all possible combinations of zeroes and ones for the variables, we may be able to prove that both sides of the equal sign always take on identical values.

(b) (Algebra) By manipulation using known identities, we may be able to prove that both sides of the equal sign are equal to the same expression.

The brute force method is guaranteed to always work, but the algebraic method is often quicker if you can see how to make the algebra work out. Let's see how these methods work.

To verify $x + x = x$ by brute force, note first that when $x = 0$ the left side is $0 + 0$, which is 0, and second that when $x = 1$ the left side is $1 + 1$, which is 1.

We now use algebra to verify $(x + y)(x + z) = x + yz$ using the identities we mentioned above. We leave it to you to indicate which of the above identities is used at each step.

$$\begin{aligned}
(x + y)(x + z) &= x(x + z) + y(x + z) \\
&= (x + z)x + (x + z)y \\
&= xx + zx + xy + zy \\
&= x + xz + xy + yz \\
&= x + xy + yz \\
&= x + yz.
\end{aligned}$$

Of course, one would not normally write out all the steps. It would probably be shortened to

$$(x + y)(x + z) = x + xy + xz + yz = x + yz,$$

just as we take shortcuts in ordinary algebra.

Example 2.18 Truth tables Two-line notation is a convenient way of proving identities by brute force when we modify it slightly: Instead of listing one function with domain $\{0,1\}^n$ and codomain $\{0,1\}$, we list several. This gives a way of building up various expressions. *Truth tables* are a modification of this: rows and columns are interchanged. Here is the truth table for the basic operations we introduced earlier.

x	y	xy	$x+y$	$x \oplus y$	x'
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

Let's use truth tables to prove $(xy)' = x' + y'$. Here's the complete proof (the $(xy)'$ and $x' + y'$ columns are equal):

x	y	xy	$(xy)'$	x'	y'	$x' + y'$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

□

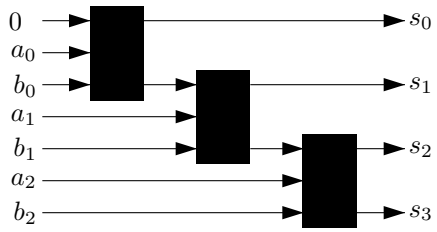
Two important identities that we have not mentioned in full generality are *DeMorgan's laws*:

$$(x \cdot y \cdot \cdots \cdot z)' = x' + y' + \cdots + z', \tag{2.3}$$

$$(x + y + \cdots + z)' = x' \cdot y' \cdot \cdots \cdot z'. \tag{2.4}$$

We prove the first by a modification of brute force and leave the second for you. The product on the left side of (2.3) is 1 if and only if $x = y = \cdots = z = 1$. Thus the left side of (2.3) is 0 if and only if $x = y = \cdots = z = 1$. The right side of (2.3) is 0 if and only if $x' = y' = \cdots = z' = 0$, which is equivalent to $x = y = \cdots = z = 1$. Thus, each side of (2.3) is 0 precisely when $x = y = \cdots = z = 1$. Consequently both sides are 1 for all other values of $x, y, \ldots, z$. This completes the proof.

Example 2.19 An adder One of the basic operations in a computer is addition. We can imagine designing a device to add two numbers the way we learned to do it in school: First add the digits in the unit's position, record the unit's digit of the sum and remember a carry digit (no carry is a carry digit of zero). Second, add the digits in the ten's position and the carry digit, record the unit's digit of the sum as the ten's digit of the answer and remember a carry digit. And so on. Of course, in a computer we work with binary instead of base ten. For one possible design of an adder, we need a device that takes in three binary digits and produces a sum unit's digit and a carry digit. Represent such a device by a black box with inputs on the left, and outputs on the right so that the top output is the sum unit's digit and the bottom output is the carry digit. To add two k digit binary numbers, we can combine k of these boxes; for example, here's how we can add the two 3 digit binary numbers $a_2a_1a_0$ and $b_2b_1b_0$ to get a sum $s_3s_2s_1s_0$:



We need to specify the outputs of our black box as Boolean functions of the inputs. One way to do this is with a truth table. Another way to do it is by writing down the equations. They are shown in Figure 2.1, with x, y and z as inputs and s and c as the sum and carry, respectively. (You

x	y	z	s	c
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$\begin{aligned} s &= x \oplus y \oplus z \\ &= x'y'z + x'yz' + xy'z' + xyz \end{aligned}$$

$$c = xy + xz + yz$$

Figure 2.1 The truth table and algebraic representations of a three bit binary adder. The sum of the bits x, y and z is the two bit binary number cs.

should be able to derive the truth table from the definition of s and c, but you may not see how to write down the algebraic equations.)

What form should we use—truth table or algebraic equations? The answer depends on what we plan to do with the function. If we plan to manipulate it, then an algebraic form is usually best. If we plan to use it in some computer aided circuit design program, then whatever form the program requires is best. □

How can we convert a truth table to an algebraic expression? One method is *disjunctive normal form*. A Boolean function is in disjunctive normal form if it is a sum (OR) of products (ANDs) with each factor in each product being either a variable or its complement. For example, the functions $x'y'z+x'yz'+xy'z'+xyz$ and $xy+xz+yz$ in Figure 2.1 are in disjunctive normal form, but $x\oplus y\oplus z$ is not.

In Example 7.3 (p. 200) we'll prove by induction that every Boolean function can be written in disjunctive normal form. We now give a different proof by showing how to convert the truth table for a function into disjunctive normal form. Let the function be $f(x_1, x_2, \ldots, x_n)$ Our disjunctive normal form will contain one term for each row of the truth table in which $f = 1$. Each term will be a product of n factors. If the row says $x_i = 1$, then x_i appears in the product; otherwise, x_i' appears in the product. For example, look at the function s in Figure 2.1. There are four rows in the truth table with $s = 1$. The first such row has $x = y = 0$ and $z = 1$, which gives us the first term, $x'y'z$, in our disjunctive normal form for s.

It is rather easy to prove that the truth table, T', of the resulting disjunctive normal form, f, equals the truth table, T, it was derived from. A row of T' will be 1 if and only if f is 1, which happens if and only if some term of f is 1. On the other hand, a term of f is 1 if and only if $x_1, \ldots, x_n$ take on the values that led to that term. Those values were associated with a row in T for which the function is 1.

Let's return to Figure 2.1 and look at a disjunctive normal form for c. From the truth table we have

$$c = x'yz + xy'z + xyz' + xyz, \tag{2.5}$$

which is not the same as the disjunctive normal form $c = xy + xz + yz$ given earlier. How can this be? There need not be a unique disjunctive normal form for a function. Is one form better than the other? Yes, since it has fewer and simpler terms than (2.5), $c = xy + xz + yz$ requires less hardware to implement as a disjunctive normal form than does (2.5). How can we find the "best" disjunctive normal form? This is a hard question—even defining "best" may be difficult.

There are other algebraic forms besides disjunctive normal form which are important. With current computer chip technology, the most important is "NAND," which is defined by

$$\text{NAND}(x_1, x_2, \ldots, x_n) = (x_1 x_2 \cdots x_n)', \quad n \geq 1;$$

that is, "NOT the AND of the x_i's." "NAND" is a contraction of "NOT AND". By one of DeMorgan's laws,

$$\text{NAND}(x_1, x_2, \ldots, x_n) \;=\; x_1' + x_2' + \cdots + x_n'. \tag{2.6}$$

It is a simple matter to prove that every Boolean function can be built up out of NANDs: We will do this by proving that every disjunctive normal form can be implemented by using NANDs. Let the disjunctive normal form be $f = p_1 + p_2 + \cdots + p_m$, where $p_i = u_{i,1}u_{i,2}\cdots u_{i,k_i}$ and each u is an input variable or its complement. To complete the proof it suffices to note that

$$\begin{aligned} f &= \text{NAND}(p_1', \ldots, p_m') && \text{by (2.6) and } p'' = p; \\ p_i' &= \text{NAND}(u_{i,1}, \ldots, u_{i,k_i}) && \text{by the definition of NAND;} \\ x_t' &= \text{NAND}(x_t). \end{aligned}$$

Thus f is a NAND of NANDs of $u_{i,j}$'s and each $u_{i,j}$ is a variable or its NAND. For example, from Figure 2.1 we have

$$\begin{aligned} c \;&=\; xy + xz + yz \;=\; \text{NAND}\Big(\text{NAND}(x,y), \text{NAND}(x,z), \text{NAND}(y,z)\Big) \\ s \;&=\; x'y'z + \cdots \;=\; \text{NAND}\Big(\text{NAND}(\text{NAND}(x), \text{NAND}(y), z), \ldots\Big). \end{aligned}$$

Does this direct translation of disjunctive normal form give us the simplest representation by NANDs? (Of course, there may be several disjunctive normal forms, so we would presumably start with the "simplest.") As you may suspect, the answer is "No." In fact, the question is not even well posed. One issue that arises immediately is what about repeated expressions; e.g., if x_1' appears several times, must we compute $\text{NAND}(x_1)$ several times, or may we just compute it once and reuse it? We may usually reuse it, but there are sometimes hardware constraints that do not allow us to. This is much like the problem of computing any complicated algebraic function in which some subexpression appears more than once: If you are clever, you will only compute that subexpression once. We'll not pursue the complex problem of representation by NANDs.

Exercises

2.4.1. Using only the identities in the text, prove $x(x + y) = x$ by algebra. (This is the absorbtion law for multiplication.)

2.4.2. Prove that $x \oplus y = x' \oplus y'$.

2.4.3. The distributive law for multiplication, $\cdot$, over addition, $+$, states that $x \cdot (y + z) = x \cdot y + x \cdot z$. State the following distributive laws. If the law is true, prove it. If the law is false, give a counterexample. *Hint.* Truth tables can be used to find proofs and counterexamples.

(a) addition, $+$, over multiplication, $\cdot$;
(b) multiplication, $\cdot$, over exclusive or, $\oplus$;
(c) addition, $+$, over exclusive or, $\oplus$;
(d) exclusive or, $\oplus$ over multiplication, $\cdot$.

2.4.4. Define $\text{NOR}(x, y, \ldots, z) = (x + y + \cdots + z)'$. Prove that every Boolean function can be expressed using NORs.

2.4.5. Write each of the following in disjunctive normal form. Try to obtain as simple a disjunctive normal form as possible.

(a) $(x \oplus y)(x + y)$.
(b) $(x + y) \oplus z$.
(c) $(x + y + z) \oplus z$.
(d) $(xy) \oplus z$.

2.4.6. Let $f(x, y, z)$ equal z unless $x = y$, in which case $f(x, y, z) = x$. Write $f(x, y, z)$ in disjunctive normal form.

2.4.7. Let $f(x, y, z, w)$ equal w unless $x = y = z$, in which case $f(x, y, z, w) = x$. Write $f(x, y, z, w)$ in disjunctive normal form. Try to obtain as simple a form as you can.

2.4.8. Let $f(x, y, z, w)$ equal zw if $x = y$ and $z + w$ otherwise. Write $f(x, y, z, w)$ in disjunctive normal form. Try to obtain as simple a form as you can.

Notes and References

Further discussion of Boolean functions and circuit design at the level of this text can be found in the texts by Gill [1] and McEliece et al. [2].

1. Arthur Gill, *Applied Algebra for the Computer Sciences*, Prentice-Hall (1976).
2. Robert J. McEliece, Robert B. Ash and Carol Ash, *Introduction to Discrete Mathematics*, Random House (1989).