CHAPTER 11

*Generating Function Topics

Introduction

The four sections in this chapter deal with four distinct topics: systems of recursions, exponential generating functions, Pólya's Theorem and asymptotics estimates. These sections can be read independently of one another. The section on "asymptotic" estimates refers to formulas in earlier sections of the chapter, but there is no need to read the section containing the formula.

"Systems of recursions," as you might guess, deals with the creation and solution of sets of simultaneous recursions. These can arise in a variety of ways. One source of them is algorithms that involve interrelated recursive subroutines. Another source is situations that can be associated with grammars. General context free grammars can lead to complicated recursions, but regular grammars (which are equivalent to finite state automata) lead to simple systems of recursions. We limit our attention to these simple systems.

Exponential generating functions are very much like ordinary generating functions. They are used in situations where things are labeled rather than unlabeled. For example, they are used to study partitions of a set because each of the elements in the set is different and hence can be thought of as a labeled object. We will briefly study the Rules of Sum and Product for them as well as a useful consequence of the latter—the Exponential Formula.

Burnside's Lemma (Theorem 4.5 (p. 112)) is easily seen to apply to the situation in which the objects being counted have "weights". As a result, we can introduce generating functions into the study of objects with symmetries. This "weighted Burnside lemma" has a variety of important special cases. We will study the simplest, and probably most important one—Pólya's theorem.

Suppose we are studying some sequence of numbers a_n and want to know how the sequence behaves when n is large. Usually it grows rapidly, but we want to know more than that—we want a relatively simple formula that provides some sort of estimate for a_n. Stirling's formula (Theorem 1.5 (p. 12)) is an example of such a formula. This subject is referred to as *asymptotics* or *asymptotic analysis*. Because the proofs of results in this area require either messy estimates, a knowledge of complex variables or both, we will not actually prove the estimates that we derive.

11.1 Systems of Recursions

So far we've dealt with only one recursion at a time. Now we look at ways in which systems of recursions arise and adapt our methods for a single recursion to solving simple systems of recursions. The adaptation is straightforward—allow there to be more than one recursion. As usual, examples are the best way to see what this is all about.

Example 11.1 Counting Batcher sort comparators Let's study the Batcher sort. As with our study of merge sorting in Example 10.4 (p. 278), we'll limit the number of things being sorted to a power of 2 for simplicity. We want to determine b_k, the number of comparators in a Batcher sort for 2^k things. We'll rewrite the Batcher sorting algorithm in Section 8.3.2 to focus on the number of things being sorted and number of comparators. The comments indicate the contributions to the recursion.

```
BSORT(2^k things)                       /* uses b_k comparators */
     If k = 0, Return                   /* b_0 = 0 */
     BSORT(2^{k-1} things)              /* b_k = b_{k-1} */
     BSORT(2^{k-1} things)              /*     +b_{k-1} */
     BMERGE(2^k things)                 /*     +m_k */
     Return
End

BMERGE(2^k things)                      /* uses m_k comparators */
     If k = 0, Return                   /* m_0 = 0 */
     End if
     If k = 1,
          one Comparator and Return     /* m_1 = 1 */
     BMERGE2(2^k things)                /* m_k = t_k */
     2^{k-1} - 1 Comparators            /*     +2^{k-1} - 1 */
     Return
End

BMERGE2(2^k things)                     /* uses t_k comparators */
     BMERGE(2^{k-1} things)             /* t_k = m_{k-1} */
     BMERGE(2^{k-1} things)             /*     +m_{k-1} */
     Return
End
```

Note that since BMERGE2(2^k things) is never called for $k < 2$, we can define t_0 and t_1 arbitrarily.

How should we choose the values of t_0 and t_1? In the end, it doesn't really matter because the answers will be unaffected by our choices. On the other hand, how we choose these values can affect the amount of work we have to do. There are two rules of thumb that can help in making such choices:

- Choose the initial values so as to minimize the number of special cases of the recursion. (For example, the recursion below has two special cases for m_k.)
- Choose simple values like zero.

We'll set $t_0 = 0$ and $t_1 = 2m_0 = 0$.

Copying the recursions from the comments in the code we have

$$
\begin{aligned}
b_k &= \begin{cases} 0 & \text{if } k=0, \\ 2b_{k-1}+m_k & \text{if } k>0; \end{cases} \\
m_k &= \begin{cases} 0 & \text{if } k=0, \\ 1 & \text{if } k=1, \\ t_k+2^{k-1}-1 & \text{if } k>1; \end{cases} \\
t_k &= \begin{cases} 0 & \text{if } k=0, \\ 2m_{k-1} & \text{if } k>0. \end{cases}
\end{aligned}
\qquad 11.1
$$

We now apply our six steps for solving recursions (p. 277), allowing more than one recursion and more than one resulting equation. Let $B(x)$, $M(x)$ and $T(x)$ be the generating functions. The result of applying the first four steps to (11.1) is

$$
\begin{aligned}
B(x) &= 2xB(x)+M(x), \\
M(x) &= x+\sum_{k\ge 2} t_k x^k + \sum_{k\ge 2}(2^{k-1}-1)x^k \\
&= x+\sum_{k\ge 0} t_k x^k + \sum_{k\ge 1}(2^{k-1}-1)x^k \\
&= x+T(x)+\frac{x}{1-2x}-\frac{x}{1-x}, \\
T(x) &= 2xM(x).
\end{aligned}
$$

We now carry out Step 5 by solving these three linear equations in the three unknowns $B(x)$, $M(x)$ and $T(x)$. From the first equation, we have

$$B(x) = \frac{M(x)}{1-2x}. \qquad 11.2$$

Combining the equations for $M(x)$ and $T(x)$, we have

$$M(x) = x+2xM(x)+\frac{x}{1-2x}-\frac{x}{1-x}.$$

Solving this for $M(x)$ and substituting the result (11.2), we obtain

$$B(x) = \frac{x}{(1-2x)^2}+\frac{x}{(1-2x)^3}-\frac{x}{(1-x)(1-2x)^2}.$$

Our formula for $B(x)$ can be expanded using partial fractions. We won't carry out the calculations here except for noting that we can rewrite this as

$$B(x) = \frac{x}{(1-2x)^3}-\frac{x}{(1-2x)^2}+\frac{2x}{1-2x}-\frac{2x}{1-x}.$$

The result is

$$b_k = 2^{k-2}(k^2-k+4)-1. \qquad 11.3$$

How does this result compare with the upper bound on the number of comparisons in merge sort that we found in Example 10.4? There we obtained an upper bound of $n\log_2 n$ and here we obtained an actual value that is close to $\frac{1}{2}n(\log_2 n)^2$. Thus Batcher sort is a poor software sorting algorithm. On the other hand, it is a much better network sorting algorithm than the only other one we studied, the brick wall. □

In the next example we will work with a set of linked recursions in the context of a directed graph or, equivalently, a finite state automaton.

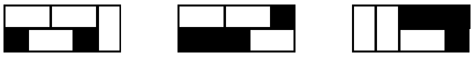

Figure 11.1 Three ways to place nonoverlapping dominoes on 2 by 5 boards. A domino is indicated by a white rectangle and the board is black.

Example 11.2 Rectangular arrays, digraphs and finite automata

Imagine a k by n array of squares. We would like to place dominoes on the array so that each domino covers exactly two squares and the dominoes do not overlap. Not all squares need be covered. Thus there may be any number of dominoes from none to $nk/2$. Let $a_n(k)$ be the number of ways to do this. Figure 11.1 shows some ways to place dominoes on a 2 by 5 board.

We can work out a recursion for $a_n(1)$ quite easily. Let's do it. When $n > 1$, the number of arrangements with the last square empty is $a_{n-1}(1)$ and the number of arrangements with a domino in the last square is $a_{n-2}(1)$. This is the same as the recursion for the Fibonacci numbers, but the initial conditions are a bit different. (What are they?) After some calculation, you should be able to get $a_n(1) = F_{n-1}$.

There is a quicker way to get $a_n(1)$. If we replace each empty square with a "0" and each domino with "10", we obtain an n-long string of zeroes and ones ending in zero and having no adjacent ones. Conversely any such string of zeroes and ones gives rise to a placement of dominoes. By stripping off the rightmost zero in the string, we have that $a_n(1) = F_{n-1}$.

For $k > 1$, $a_n(k)$ is much more complicated. Try to write down a recursion for $a_n(2)$—or calculate it in any other manner.

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

We will show how to use a directed graph to produce our patterns of dominoes. The graph can be regarded as a finite state machine or, more specifically, a nondeterministic finite automaton (Section 6.6). We will show how to associate generating functions with such digraphs.

Our machine (digraph) has four states (vertices), which we call *clear, first, second* and *both.* We imagine moving across the $2\times n$ array, one column at a time from left to right, placing dominoes as we reach a column. When we decide to place a domino in a horizontal position starting in column j of the array, we are covering a square in column $j+1$ that we haven't yet reached. We call this covering of an unreached square a "commitment." At the next step we must take into account any commitment that was made in the previous step. Our states keep track of the unsatisfied commitments; that is, if we are filling the jth column, the commitments made at column $j-1$:

(a) *clear* means there are no commitments to take into account;

(b) *first* means we made a commitment by placing a domino in the first row of column $j-1$ which runs over into column j;

(c) *second* means we made a commitment by placing a domino in the second row;

(d) *both* means we made commitments by placing dominoes in both rows.

Using the letters, c, f, s and b, the sequences of states for the columns in Figure 11.1 are $fsfcc$, $fcfsc$ and $ccscc$, respectively. The columns to which the commitments are made are 2 through 6. No commitments are made to column 1 because no dominoes are placed before column 1. If we wanted to reflect this fact in our sequences, we could add an entry of c for column 1 at the beginning of each sequence. Note that all of these strings end with a c because no dominoes hang over the right end of the board.

There is an edge in our digraph from state x to state y for each possible way to get from state x at column j of the array to state y at column $j+1$ of the array. For example, we can go from *clear* to *clear* by placing no dominoes in column j or by placing one vertical domino in the column. Thus there are two loops at *clear.* Since we cannot go from *first* to *both*, there are no edges from *first* to *both*. The following table summarizes the possibilities. An entry $m_{x,y}$ in position (x, y) is the number of edges from state x to state y.

	clear	*first*	*second*	*both*
clear	2	1	1	1
first	1	0	1	0
second	1	1	0	0
both	1	0	0	0

To complete our picture, we need the initial and accepting states. From the discussion at the end of the previous paragraph, you should be able to see that our initial state should be *clear* and that there should be one accepting state, which is also *clear*.

Let c_n be the number of ways to reach the state *clear* after taking n steps from the starting state, *clear*. This means that no dominoes hang over into column $n+1$. Note that c_n is the number of ways to place dominoes on a $2 \times n$ board. Define f_n, s_n and b_n in a similar way according to the state reached after n steps from the starting state. Let $C(x)$, etc., be the corresponding generating functions.

We are only interested in c_n (or $C(x)$), so why introduce all these other functions? The edges that lead into a state give us a recursion for that state, for example, looking down the column labeled *clear*, we see that

$$c_{n+1} \;=\; 2c_n + f_n + s_n + b_n \qquad 11.4$$

for $n \geq 0$. Thus, when we study c_n this way, we end up needing to look at the functions f_n, s_n and b_n, too.

In a similar manner to the derivation of (11.4),

$$\begin{aligned} f_{n+1} &= c_n + s_n, \\ s_{n+1} &= c_n + f_n, \\ b_{n+1} &= c_n, \end{aligned} \qquad 11.5$$

for $n \geq 0$. The initial conditions for the recursions (11.4) and (11.5) are the values at $n = 0$. Since the initial state, *clear*, is the only state we can get to in zero steps,

$$c_0 = 1 \quad \text{and} \quad f_0 = s_0 = b_0 = 0.$$

To see how much work these recursions involve, use them to find c_8, the number of ways to place dominoes on a 2 by 8 board. Can we find an easier way to calculate c_n; for example, one that does not require the values of f, s and d as well? Yes.

We begin by converting the recursions into a set of linked generating functions using our method for attempting to solve recursions. Multiplying both sides of the equations (11.4) and (11.5) by x^{n+1}, summing over $n \geq 0$ and using the initial conditions, we obtain

$$\begin{aligned} C(x) &= x(2C(x) + F(x) + S(x) + B(x)) + 1 \\ F(x) &= x(C(x) + S(x)) \\ S(x) &= x(C(x) + F(x)) \\ B(x) &= xC(x). \end{aligned}$$

We have four linear equations in the four unknowns $C(x)$, $F(x)$, $S(x)$ and $B(x)$. Let's solve them for $C(x)$. Subtracting the second and third equations, we get $F(x) - S(x) = x(S(x) - F(x))$ for all x and so $S(x) = F(x)$. Thus $F(x) = xC(x) + xF(x)$ and so $F(x) = S(x) = xC(x)/(1 - x)$. Substituting this and $B(x) = xC(x)$ into the first equation gives us

$$C(x) \;=\; 2xC(x) + \frac{2x^2C(x)}{1-x} + x^2C(x) + 1.$$

With a bit of algebra, we easily obtain

$$C(x) \;=\; \frac{1-x}{1-3x-x^2+x^3}. \qquad 11.6$$

A method of obtaining this by working directly with the table, thought of as a matrix

$$M \;=\; \begin{pmatrix} 2 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \qquad 11.7$$

is given in Exercises 11.1.7.

We can use partial fractions to determine c_n, but before doing so, we'd like to note that (11.6) gives us an easier recursion for calculating c_n: From (11.6) we have $C(x) = 1-x+(3x+x^2-x^3)C(x)$. Looking at the coefficient of x^n on both sides, we have

$$c_0 = 1, \qquad c_1 = -1+3c_0 = 2, \qquad c_2 = 3c_1 + c_0 = 7$$

and

$$c_n \;=\; 3c_{n-1} + c_{n-2} - c_{n-3} \quad \text{for} \quad n > 2. \qquad 11.8$$

Use this recursion to find c_8 and compare the amount of work with that when you were using (11.4) and (11.5). It's also possible to derive (11.8) by manipulating (11.4) and (11.5) without using generating functions. (Try doing it.)

To use partial fractions, we must factor the denominator of (11.6):

$$1-3x-x^2+x^3 \;=\; (1-px)(1-qx)(1-rx).$$

Unfortunately, the cubic does not factor with rational p, q or r. Using numerical methods we found that $p = 3.21432\ldots$, $q = .46081\ldots$ and $r = -.67513\ldots$. By partial fractions, $c_n = Pp^n + Qq^n + Rr^n$, where $P = .66459\ldots$, $Q = .07943\ldots$ and $R = .25597\ldots$. Thus a_n is the integer closest to Pp^n. Since we know p and P only approximately, we can't get c_n exactly for large n this way. Instead, we'd use the recursion (11.8) to get exact values for c_n. On the other hand, the recursion gives us no idea about how fast c_n grows, so we'd use our partial fraction result for this sort of information. For example, the result says that, in some sense, the average number of choices per column is about p since if there were exactly p choices per column, there would be p^n ways to place dominoes in n columns. □

Example 11.3 Binary operations Let $\wedge$ denote exponentiation; e.g., $3 \wedge 2 = 9$. The interpretation of $2 \wedge 3 \wedge 2$ is ambiguous. We can remove the ambiguity by using parentheses:

$$(2 \wedge 3) \wedge 2 = 8 \wedge 2 = 64$$
$$2 \wedge (3 \wedge 2) = 2 \wedge 9 = 128.$$

Sometimes we get the same answer from different parenthesizations; e.g., $2 \wedge 2 \wedge 2 = 16$, regardless of parentheses.

Let's consider the possible values of

$$0 \wedge 0 \wedge \ldots \wedge 0. \qquad 11.9$$

Unfortunately, $0 \wedge 0$ is not defined for the real numbers. For the purpose of this example, we'll define $0 \wedge 0 = 1$. The only other powers we need are well defined. In summary

$$0 \wedge 0 = 1, \quad 0 \wedge 1 = 0, \quad 1 \wedge 0 = 1 \quad \text{and} \quad 1 \wedge 1 = 1. \qquad 11.10$$

You should be able to show by induction on the length of the string that the only possible values for (11.9) are 0 and 1, regardless of how the expression is parenthesized.

How many of the parenthesizations of (11.9) give the expression the value 0 and how many give it the value 1?

If there are n zeroes present, let z_n be the number of ways to obtain the value 0 and let w_n be the number of ways to obtain 1. We can write down the generating functions by using the Rules of

Sum and Product: A zero is produced if $n = 1$ OR, by (11.10), if the left string is a zero AND the right string is a one. By (11.10), the other three combinations give a one. Thus

$$\begin{aligned} Z(x) &= x + Z(x)W(x) \\ W(x) &= Z(x)Z(x) + W(x)Z(x) + W(x)W(x). \end{aligned} \tag{11.11}$$

These equations are more complicated than our earlier ones because they are not linear. In general, we cannot hope to solve such equations; however, these can be solved. Try to do so before you continue.

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

Let $T(x) = W(x) + Z(x)$, the total number of parenthesizations without regard to value. Using algebra on (11.11) or, more simply, using a direct combinatorial argument, you should be able to show that

$$T(x) = x + T(x)T(x).$$

The solution to this equation is

$$T(x) = \frac{1 - \sqrt{1-4x}}{2}, \tag{11.12}$$

where the minus sign was chosen on the square root because $T(0) = t_0 = 0$. We now rewrite $Z(x) = x + Z(x)W(x)$ as

$$Z(x) = x + Z(x)\big(T(x) - Z(x)\big) = x + T(x)Z(x) - Z(x)^2. \tag{11.13}$$

Solving this quadratic for $Z(x)$:

$$Z(x) = \frac{T(x) - 1 + \sqrt{\big(1 - T(x)\big)^2 + 4x}}{2}, \tag{11.14}$$

where the root with the plus sign was chosen because $Z(0) = 0$.

Of course, we can substitute (11.12) into (11.14) to obtain the explicit formula for $Z(x)$. Unfortunately, we are unable to extract a nice formula for z_n from the result.

What can be done? At present, not much; however, in Example 11.31 (p. 351), we will obtain estimates of z_n for large n.

Note that $T(x) = B(x)$, the generating function for unlabeled full binary RP-trees. We obtained a differential equation for B in Example 10.8 (p. 284). This led to a simple recursion. There are general techniques for obtaining such differential equations and hence recursions, but they are beyond the scope of this text. We merely remark that it is possible to obtain a recursion for z_n that requires much less work than would be involved in simply using the recursion that is obtained by extracting the coefficient of x^n in (11.13). $\square$

Exercises

*11.1.1. Derive (11.8) directly from (11.4) and (11.5) without using generating functions.

11.1.2. Redo Exercise 10.2.4 (p. 280) using the directed graph method of Example 11.2. Which way was easier?
Hint. Here's one way to set it up. Introduce three vertices to indicate the present digit: *zero*, *one* and *two*. You can introduce a bad vertex, too, or make certain digits impossible. Since you can end at any digit, you'll want to add generating functions together to get your answer.

11.1.3. Let a_n be the number of ways to place dominoes on a 3 by n board with no blank spaces. Note that $a_n = 0$ when n is odd. Let $A(x)$ be the generating function.

(a) Prove that $A(x) = (1 - x^2)/(1 - 4x^2 + x^4)$.

(b) Show that $a_n = 4a_{n-2} - a_{n-4}$ when $n \geq 4$.

*(c) Prove the previous recursion without the use of generating functions.

11.1.4. How many n-long sequences of zeroes ones and twos are there with no adjacent entries equal?

11.1.5. Let $\mathcal{L}$ be a set of strings and let a_n be the number of strings in $\mathcal{L}$ of length n. Suppose that we are given a nondeterministic finite automaton for recognizing precisely the strings in $\mathcal{L}$. Describe a method for using the automaton to get the generating function for the a_n's.

11.1.6. Let $a_{n,j}(2)$ be the number of ways to place exactly j dominoes on a 2 by n board. Extend the finite machine approach of Example 11.2 so that you can calculate $\sum_{n,j} a_{n,j}(2)x^n y^j$.
Hint. Put on the edges of the digraph information about the number of dominoes added to the board. Use it to write down the equations relating the generating functions.

11.1.7. Given a finite machine as in Example 11.2, let $m_{x,y}$ be the number of edges from state x to y. This defines a matrix M of (11.7).

(a) Show that $m_{x,y}^{(n)}$, the (x,y) entry in M^n, is the number of ways to get from x to y in exactly n steps.

(b) Let $\mathbf{a}$ be a row vector with $a_k = 1$ if k is an accepting state and $a_k = 0$ otherwise. Define $\mathbf{i}$ similarly for the initial state. Show that $\mathbf{i}M^n\mathbf{a}^t$ is the number of ways to get from the initial state to an accepting state in exactly n steps. (We use $\mathbf{a}^t$ for the column vector which is the transpose of the row vector $\mathbf{a}$.)

(c) Conclude that the generating functions in Example 11.2 and in Exercise 11.1.5 have the form $\mathbf{i}(I - xM)^{-1}\mathbf{a}^t$, where I is the identity matrix.

(d) Extend the definition of M to include the previous exercise.

11.1.8. Let a_n be the number of ways to distribute unlabeled balls into boxes numbered 1 through n such that for each j with $1 \leq j < n$, the number of balls in boxes j and $j+1$ together is at most 2.

(a) Construct a directed graph that can be used to calculate the generating function for a_n.
Hint. Let the vertices (states) be the number of balls in a box.

(b) Obtain a set of linked equations for the various generating functions.

(c) Show that $\sum a_n x^n = (1 + x - x^2)/(1 - 2x - x^2 + x^3)$.

(d) Restate the solution in matrix terms using the previous exercise.

(e) Find the roots of $r^3 - 2r^2 - r + 1 = 0$ numerically and use partial fractions to determine the value of a_n.

(f) Let $b_{n,k}$ be the number of ways to distribute k balls into n boxes subject to our adjacency condition. Using the idea in Exercise 11.1.6, determine the generating function for $b_{n,k}$.

11.1.9. How one approaches a problem can be very important. Obviously, the wrong attack on a problem may result in no solution. Less obvious is the fact that one approach may involve much less work than another. Everyone sometimes fails to find the easiest approach. You can probably find examples of this in the way you've solved some homework problems. Here's another brief example. How many n long sequences of zeroes and ones contain the pattern $p = 010101011$?

(a) One approach is to use two of the tools we've developed: designing finite automata for accepting sequences with certain patterns and obtaining generating functions from automata. This would require an automaton with at least ten states. Draw such an automaton for p and write down the family of associated equations.

(b) We'll look at a simpler approach. Let a_n be the number of n long strings that do *not* contain the desired pattern p. By considering what happens when something is added to the end of an $n-1$ long pattern, show that $2a_{n-1} = a_n + c_n$ for $n > 0$, where c_n is the number of n long strings $s_1, \ldots, s_n$ containing p but with p not contained in $s_1, \ldots, s_{n-1}$. Also show that $c_n = 0$ for $n < 9$, the length of p.

(c) Show that $c_n = a_{n-9}$ and conclude that $A(x) = (1 - x + x^9)^{-1}$.
Hint. If your proof also works for 001001001, it is not quite correct.

(d) Generalize the previous result as much as you can.

(e) Show that the finite automata approach might sometimes be better by giving an automaton for recognizing the strings that contain the pattern 001001001.

11.1.10. The situation we studied in Example 11.3 can be generalized in various ways. In this exercise you will study some possibilities.

(a) Suppose we have a set A of symbols and a binary operation $\circ$ on A. This means that for all $s, t \in A$ the value of $s \circ t \in A$ is given. Consider the "product" $b \circ b \circ \ldots \circ b$. We want to know how many ways each of the elements $s \in A$ can arise by parenthesizing this product. Describe carefully how to obtain the equations relating the generating functions from the "multiplication" table for the operation $\circ$.

(b) We can change the previous situation by letting $\circ$ be a k-ary operation. For example, if it is a ternary operation, there is no way to make sense of either the expression $b \circ b$ or the expression $b \circ b \circ b \circ b$. On the other hand, we have three parenthesizations for the 5-long case:

$$(b \circ b \circ b) \circ b \circ b, \quad b \circ (b \circ b \circ b) \circ b \ \text{ and } \ b \circ b \circ (b \circ b \circ b).$$

Again, describe how to construct equations relating the generating functions.

11.2 Exponential Generating Functions

When we use ordinary generating functions, the parts we are counting are "unlabeled." It may appear at first sight that this was not so in all the applications of ordinary generating functions. For instance, we had sequences of zeroes and ones. Isn't this labeling the positions in a sequence? No, it's dividing them into two classes. If they were labeled, we would require that each entry in the sequence be different; that is, *each label would be used just once*. Well, then, what about placing balls into labeled boxes? Yes the boxes are all different, but the parts we are counting in our generating functions are the *unlabeled* balls, not the boxes. The boxes simply help to form the structure.

In this section, we'll use exponential generating functions to count structures with labeled parts. What we've said is rather vague and may have left you confused. We need to be more precise, so we'll look at a particular example and then explain the general framework that it fits into.

Recall the problem of counting unlabeled full binary RP-trees by number of leaves. We said that any such tree could be thought of as either a single vertex OR an ordered pair of trees. Let's look at the construction of this ordered pair a bit more closely. If the final tree is to have n leaves, we first choose some number k of leaves and construct a tree with that many leaves, then we construct a tree with $n - k$ leaves as the second member of the ordered pair. Thus there is a three step procedure:

1. Determine the number of leaves for the first tree (and hence also the second), say k.
2. Construct the first tree so that it contains k leaves.
3. Construct the second tree so that it contains $n-k$ leaves.

Now let's look at what happens if the leaves are to be labeled; i.e., there is a bijection from the n leaves to some set N of n labels. (Usually we have $N = \underline{n}$, but this need not be so.) In this case, we must replace our first step by a somewhat more complicated step and modify the other two steps in an obvious manner:

1′. Determine a subset K of N which will become the labels of the leaves of the first tree.
2′. Construct the first tree so that its leaves use K for labels.
3′. Construct the second tree so that its leaves use $N-K$ for labels.

Note that the number of ways to carry out Steps 2′ and 3′ depend only on $|N|$ and $|K|$, not on the actual elements of N and K. This is crucial for the use of exponential generating functions. Because of this, it is convenient to split Step 1′ into two steps:

1′a. Determine the number of leaves for the first tree (and hence also the second), say k.
1′b. Determine a subset K of N with $|K| = k$ to be the leaves of the first tree.

Let b_n be the number of unlabeled full binary RP-trees with n leaves and let t_n be the number of such trees except that the leaves have been labeled using some set N with $|N| = n$. For $n > 1$, our unlabeled construction gives us $\sum_k b_k b_{n-k}$, where the summation comes from the first step, the b_k from the second and the b_{n-k} from the third. Similarly, for the labeled construction, we have

$$\sum_{k=0}^{n} \binom{n}{k} t_k t_{n-k} \quad \text{for } n > 1, \tag{11.15}$$

where now the summation comes from Step 1′a and the binomial coefficient from Step 1′b.

The initial condition $t_1 = 1$ (and, if we want, $t_0 = 0$) together with (11.15) gives us a recursion for t_n. We'd like to use generating functions to solve this recursion as we did for the unlabeled case. The crucial step for the unlabeled case was the observation that we were dealing with a convolution which then led to (10.18). If this approach is to work in the labeled case, we need to be able to view (11.15) as a convolution, too.

Recall that a convolution is something of the form $\sum_k a_k c_{n-k}$. Unfortunately, (11.15) contains $\binom{n}{k}$ which is not a function of k and is not a function of $n-k$, so it can't be included in a_k or in c_{n-k}. Fortunately, we can get around this by rewriting the binomial coefficient in terms of factorials:

$$t_n = \sum_{k=0}^{n} \binom{n}{k} t_k t_{n-k} = n! \sum_{k=0}^{n} \frac{t_k}{k!} \frac{t_{n-k}}{(n-k)!}.$$

If we divide this equation by $n!$, we get a recursion for $t_n/n!$ in which the sum is a convolution. Thus, the generating function $B(x) = \sum b_n x^n$ in the unlabeled case should be replaced by the generating function $T(x) = \sum (t_n/n!) x^n$ in the labeled case. We can then proceed to solve the problem just as we did in the unlabeled case.

You may have noticed that $t_n = b_n n!$, a result which can be proved directly. So why go through all this? We did it to introduce an idea, not to solve a particular problem. So let's formulate the idea.

Definition 11.1 Exponential generating function *The* **exponential generating function** *for the sequence c is*

$$\sum_{n \ge 0} c_n (x^n/n!).$$

If $\mathcal{T}$ is some set of structures and $w(T)$ is the number of labels in T, then $\mathsf{E}_{\mathcal{T}}(x)$, the exponential generating function for $\mathcal{T}$ is $\sum_{T \in \mathcal{T}} x^{w(T)}/(w(T))!$. We abbreviate "exponential generating function" to **EGF**.

Theorem 11.1 Rule of Sum *Suppose a set $\mathcal{T}$ of structures can be partitioned into sets $\mathcal{T}_1, \ldots, \mathcal{T}_j$ so that each structure in $\mathcal{T}$ appears in exactly one $\mathcal{T}_i$. It then follows that*

$$\mathsf{E}_{\mathcal{T}} = \mathsf{E}_{\mathcal{T}_1} + \cdots + \mathsf{E}_{\mathcal{T}_j}.$$

Theorem 11.2 Rule of Product *Suppose each structure in a set $\mathcal{T}$ of structures can be constructed from an ordered partition (K_1, K_2) of the labels and some pair (T_1, T_2) of structures using the labels K_1 in T_1 and K_2 in T_2 such that:*

(i) The number of ways to choose a T_i with labels K_i depends only on i and $|K_i|$.

(ii) Each structure $T \in \mathcal{T}$ arises in exactly one way in this process.

(We allow the possibility of $K_i = \emptyset$ if $\mathcal{T}_i$ contains structures with no labels.) It then follows that

$$\mathsf{E}_{\mathcal{T}}(x) = \mathsf{E}_1(x)\mathsf{E}_2(x),$$

where $\mathsf{E}_i(x) = \sum_{n=0}^{\infty} t_{i,n} x^n/n!$ and $t_{i,n}$ is the number of ways to choose T_i with labels $\underline{n}$.

Proof: You should be able to prove the Rule of Sum. The Rule of Product requires more work. Let t_n be the number of $T \in \mathcal{T}$ with $w(T) = n$. By the assumptions,

$$t_n = \sum_{K_1 \subseteq \underline{n}} t_{1,|K_1|}\, t_{2,n-|K_1|}$$

and so

$$\frac{t_n}{n!} = \frac{1}{n!} \sum_{k=0}^{n} \binom{n}{k} t_{1,k}\, t_{2,n-k} = \sum_{k=0}^{n} \frac{t_{1,k}}{k!} \frac{t_{2,n-k}}{(n-k)!},$$

a convolution. Multiply by x^n, sum over n and rearrange to obtain $\mathsf{E}_{\mathcal{T}}(x) = \mathsf{E}_1(x)\mathsf{E}_2(x)$. $\square$

If you compare these theorems with those given for ordinary generating functions in Section 10.4, you may notice some differences.

- First, the Rule of Product was stated here for a sequence of only two choices. This is not an essential difference—you can remove the constraint at the expense of a more cumbersome statement of the theorem or you can simply iterate the theorem: divide things into two choices, then subdivide the second choice in two and so on.
- The second difference seems more substantial: There is no parallel to condition (iii) of the ordinary generating function version, Theorem 10.4 (p. 292). This is because it is implicitly built into the statement of the theorem—the EGF counts by total number of labels, so it follows that if T comes from T_1 and T_2, then $w(T_1) + w(T_2) = w(T)$.
- Finally, neither of the theorems here mentions either an infinite number of blocks in the partition (Rule of Sum) or an infinite number of steps (Rule of Product). Again, this is not a problem—infinitely many is allowed here, too.

Example 11.4 Counting derangements Recall that a derangement is a permutation with no fixed points and that D_n denotes the number of derangements on an n element set. It is convenient to say that there is one permutation of the empty set and that it is a derangement because it does not map anything to itself. By splitting off the fixed points from an arbitrary permutation, say $n-k$ of them, we have

$$n! \;=\; \sum_{k=0}^{n} \binom{n}{k} D_k. \tag{11.16}$$

We can manipulate this to obtain an EGF for D_n, but it is easier to go directly from the combinatorial argument to the Rule of Product. We'll do that now.

Let $D(x)$ be the EGF for derangements by number of things deranged. A permutation of $\underline{n}$ can be constructed by choosing some subset K of $\underline{n}$, constructing a derangement of K and fixing the elements of $\underline{n} - K$. Every permutation of $\underline{n}$ arises exactly once this way. We make three simple observations:

- The EGF for all permutations is

$$\sum_{n=0}^{\infty} n!(x^n/n!) \;=\; \frac{1}{1-x}.$$

- Since there is just one permutation fixing all the elements of a given set, the EGF for permutations with all elements fixed is $\sum x^n/n! = e^x$.
- By the Rule of Product, $1/(1-x) = D(x)e^x$ and so

$$D(x) \;=\; \frac{e^{-x}}{1-x}. \tag{11.17}$$

Note that we never needed to write down (11.16). In Example 10.7 (p. 283) and Exercise 10.3.1 (p. 291), (11.17) was used to obtain simple recursions for D_n.

The simplicity of this derivation and the ones in the examples that follow illustrate the power of the Rule of Product. □

Example 11.5 Sequences of letters How many n long sequences can be formed from A, B and C so that the number of A's in the sequence is odd and the number of B's in the sequence is odd? The labeled objects are simply the positions in the sequence. We form an ordered partition of the labels $\underline{n}$ into three parts, say (P_A, P_B, P_C). The letter A is placed in all the positions that are contained in the set P_A, and similarly for B and C. This is just the set up for the Rule of Product. If $|P_A|$ is odd, we can place the A's in just one way, while if $|P_A|$ is even, we cannot place them because of the requirement that the sequence contain an odd number of A's. Thus the EGF for the A's is

$$\sum_{k \text{ odd}} x^k/k! \;=\; (e^x + e^{-x})/2.$$

This is also the EGF for the B's. For the C's we have $\sum x^k/k! = e^x$. Thus the EGF for our sequences is

$$\left(\frac{e^x + e^{-x}}{2}\right)^2 e^x \;=\; \frac{e^{3x} + 2e^x + e^{-x}}{4},$$

and so the answer is $(3^n + 2 + (-1)^n)/4$. You might like to try to find a direct counting argument for this result.

This could also have been done with ordinary generating functions and multisection of series: Keep track of the number of A's, the number of B's and the length of the sequence using three variables in the generating function. Then use multisection twice to insure an odd number of A's and an odd number of B's. Finally, set the variables that are keeping track of the A's and B's equal to 1. We will not carry out the details. □

As noted in the last example, some problems can be done with both ordinary and exponential generating functions. In such cases, it is usually clear that one method is easier than the other. In some other problems, it is necessary to use generating functions that are simultaneously exponential and ordinary. This happens because one class of objects we're keeping track of has labels and the other class does not. Here's an example of this.

Example 11.6 Words from a collection of letters In Example 1.11 (p. 13) we considered the problem of counting strings of letters of length k, where the letters can be repeated but the number of repetitions is limited. Specifically, we used the letters in the word ERROR but insisted no letter could be used more than it appeared. We suggest that you review Example 1.11 (p. 13) and the improved methods in Examples 1.19 (p. 24) and 3.3 (p. 69), where we used the letters in ERRONEOUSNESS. Imagine letters that are labeled, each by its position in the word. Since there are three E's and only one word of any length can be built with just E's, the EGF for words of E's is $1+x+x^2/2+x^3/6$. Choose E's and R's and O's and N's and U's and S's. In this the generating function for words is

$$\left(1+x+\frac{x^2}{2}+\frac{x^3}{6}\right)^2\left(1+x+\frac{x^2}{2}\right)^3(1+x)$$
$$= 1+6x+\frac{35x^2}{2}+\frac{197x^3}{6}+\frac{265x^4}{6}+45x^5+\frac{322x^6}{9}+\frac{811x^7}{36}$$
$$+\frac{541x^8}{48}+\frac{40x^9}{9}+\frac{389x^{10}}{288}+\frac{29x^{11}}{96}+\frac{13x^{12}}{288}+\frac{x^{13}}{288},$$

where the multiplication was done by a symbolic manipulation package. Multiplying the coefficient of x^n by $n!$ gives the number of n-long words. Hence the number of 8-long words is 454,440. □

Example 11.7 Set partitions We want to count the number of partitions of a set, keeping track of the size of the set that is being partitioned and the number of blocks in the partition. Since the elements of the set are distinct, they are labeled and so we will use an EGF to count them. On the other hand, the blocks of a partition are not labeled, so it is natural to use an ordinary generating function to count blocks.

Let's start by looking at partitions with one block. There is just one such, so the EGF is $\sum_{n>0} x^n/n! = e^x - 1$.

What about partitions with two blocks? We can use the Rule of Product. In fact, the statement of the Rule of Product has built into it partitions of the set into two blocks K and L. Thus the EGF should be $(e^x-1)^2$. This is not quite right because these blocks are *ordered* but the blocks of a partition of a set are supposed to be unordered. As a result, we must divide by 2!.

You should have no trouble showing that the number of partitions of a set that have exactly k blocks has the EGF $(e^x-1)^k/k!$.

Recall that $S(n,k)$, the Stirling number of the second kind, is the number of partitions of an n element set into exactly k blocks. By the previous paragraph,

$$\sum_n S(n,k)x^n/n! \;=\; \frac{(e^x-1)^k}{k!}. \tag{11.18}$$

Let $S(0,0)=1$. It follows that

$$\sum_{n,k} S(n,k)(x^n/n!)y^k \;=\; \sum_{k=1}^{\infty}\frac{(e^x-1)^k}{k!}y^k + S(0,0) \;=\; \exp(y(e^x-1)).$$

($\exp(z)$ is another notation for e^z.) Call this $A(x,y)$.

The formula for $A(x,y)$ can be manipulated in various ways to obtain recursions, formulas and other relations for $S(n,k)$. Of particular interest is the total number of partitions of a set, which is given by $B_n = \sum_k S(n,k)$. You should be able to see that we can obtain its generating function by setting $y = 1$. Thus

$$\begin{aligned}\sum_n B_n x^n/n! \;&=\; A(x,1) \;=\; \exp(e^x - 1)\\ &=\; e^{-1}\exp(e^x) \;=\; \frac{1}{e}\sum_{i=0}^{\infty}\frac{(e^x)^i}{i!} \;=\; \frac{1}{e}\sum_{i=0}^{\infty}\frac{e^{ix}}{i!}\\ &=\; \frac{1}{e}\sum_{i=0}^{\infty}\frac{1}{i!}\sum_{n=0}^{\infty}\frac{(ix)^n}{n!} \;=\; \sum_{n=0}^{\infty}\left(\frac{1}{e}\sum_{i=0}^{\infty}\frac{i^n}{i!}\right)\frac{x^n}{n!}.\end{aligned}$$

Thus we obtain

Theorem 11.3 Dobinski's formula *The number of partitions of $\underline{n}$ is*

$$\frac{1}{e}\sum_{i=0}^{\infty}\frac{i^n}{i!}. \quad \square$$

Example 11.8 Mixed generating functions Suppose we are keeping track of both labels and unlabeled things. For example, we might count set partitions keeping track of both the size of the set and the number of blocks. We state without proof

Theorem 11.4 Mixed Generating Functions *If we are keeping track of more that one thing, some of which are labeled and some of which are unlabeled, then the Rules of Sum and Product still apply. For the Rule of Product, the labels must satisfy the conditions in Theorem 11.2 and the remaining weights must satisfy the conditions in Theorem 10.4 (p. 292).*

Returning to the set partition situation, if y keeps track of the number of blocks, then (11.18) becomes

$$y^k \sum_n S(n,k)x^n/n! \;=\; \frac{y^(e^x-1)^k}{k!}$$

and so the generating function is $\sum S(n,k)x^n y^k/n! \;=\; \exp(y(e^x-1))$. $\square$

The Exponential Formula

Before stating the Exponential Formula, we'll look at a special case.

Example 11.9 Connected graphs Let g_n be the number of simple graphs with vertex set $\underline{n}$ and let c_n the number of such graphs which are connected. Recall that a simple graph is a graph whose edge set is a subset of $\mathcal{P}_2(V)$. Since each pair of vertices is either an edge or not and there are $\binom{n}{2}$ pairs of vertices, it follows that $g_n = 2^{\binom{n}{2}}$. (This is *not* $2\binom{n}{2}$; the $\binom{n}{2}$ is an exponent.) What is the value of c_n?

This problem can be approached in various ways. We'll look at it in a way that can be generalized considerably so that we'll be able to obtain other results. The basic idea is to view a graph as being built from connected components. We must either

- figure out how a graph decomposition into connected components translates into a generating function or
- find some way to distinguish a component so we can split off one component and thus proceed in a recursive fashion.

We'll follow the latter approach. In order to distinguish a component, we will root the graph.

Let $G(x)$ and $C(x)$ be the EGFs for g_n and c_n. It will be convenient to take $g_0 = 1$ and $c_0 = 0$.

Imagine rooting the graph by choosing some vertex to be the root. There are n distinct ways to do this and so there are ng_n such graphs. Thus the generating function for the rooted graphs is $xG'(x)$.

We can construct a rooted graph by choosing a rooted component and then choosing the rest of the graph. This is just the set up for the Rule of Product. Rooted components have the EGF $xC'(x)$. The rest of the rooted graph is simply a graph and so its generating function is $G(x)$. (Note that this works even when the rest of the rooted graph is empty because we have $g_0 = 1$.) We have proved that

$$xG'(x) \;=\; \big(xC'(x)\big)\,G(x). \qquad 11.19$$

Ignoring questions of convergence, as we usually do, we can easily solve this differential equation by separation of variables to obtain

$$C(x) \;=\; \ln\!\big(G(x)\big) + A,$$

where the constant A needs to be determined. (Here's how separation of variables works in this case. We have $xdG/dx = (xdC/dx)G$ and so $dG/G = dC$, which we integrate.)

Since $C(0) = c_0 = 0$ and $G(0) = g_0 = 1$, it follows that $A = 0$. Thus we have $G(x) = \exp(C(x))$. It would be nice if this formula led to a simple method for calculating $C(x)$. This is not the case—in fact, it is easier to equate coefficients of x^n in (11.19) and rearrange it into a (rather messy) recursion for c_n. □

The formula $G = e^C$ has been generalized considerably in the research literature. Here is one form of it.

Theorem 11.5 Exponential Formula *Suppose that we have two sets $\mathcal{S}$ and $\mathcal{C}$ of labeled structures. A structure is rooted by distinguishing one its labels and calling it the root. Form all possible rooted structures from those in $\mathcal{S}$ and call the set $\mathcal{S}_r$. Do the same for $\mathcal{C}$. If the Rule of Product holds so that $\mathsf{E}_{\mathcal{S}_r} \;=\; \mathsf{E}_{\mathcal{C}_r}\mathsf{E}_{\mathcal{S}}$, then*

$$\mathsf{E}_{\mathcal{S}} \;=\; \exp\!\big(\mathsf{E}_{\mathcal{C}}\big).$$

($\exp(z)$ is another notation for e^z.) The proof is the same as that for $G = e^C$.

Example 11.10 Graphs revisited The previous example can be extended in two important directions.

More parameters: We could include more variables in our generating functions to keep track of other objects besides number of vertices. The basic requirement is that the number of such objects in our rooted graph must equal the number in its rooted component plus the number in the rest of the graph so that we still have the differential equation $x\frac{\partial G}{\partial x} = (x\frac{\partial C}{\partial x})G$. (Of course, G and C now contain other variables besides x and so the derivative with respect to x is a partial derivative.) You should be able to easily see that we still have the solution $G = \exp(C)$, either from the differential equation or from Theorem 11.5. Let's look at a couple of examples.

- We can keep track of the number of components in a graph. Let $g_{n,k}$ be the number of simple graphs with $V = \underline{n}$ that have exactly k components. The generating function is $G(x,y) = \sum_{n,k} g_{n,k}(x^n/n!)y^k$ because the vertices are labeled but the components are not. Of course, a connected graph has exactly one component. Thus $C(x,y) = C(x)y$. We have $G(x,y) = \exp(C(x)y)$. Since $\exp(C(x)) = G(x)$, it follows that

$$G(x,y) \;=\; G(x)^y. \qquad 11.20$$

What does this expression mean; that is, how should one interpret $G(x)^y$?

We can write $G(x) = 1 + xH(x)$ for some power series $H(x)$. By the binomial theorem for arbitrary powers we have

$$G(x)^y \;=\; \sum_{k=0}^{\infty} \binom{y}{k} x^k H(x)^k. \qquad 11.21$$

This expression makes perfectly good sense: We have

$$\binom{y}{k} \;=\; \frac{y(y-1)\cdots(y-k+1)}{k!},$$

and if we want a coefficient, say that of $x^n y^m$, we need only look at a finite number of terms, namely those with $k \le n$. (You may be concerned that (11.21) is really the same as $G(x,y) = \exp(yC(x))$. It is. This can be shown by formal power series manipulations.)

It may appear that (11.20) gives us something for nothing—just by knowing the number of graphs by vertices we can determine the number of graphs by both vertices and components. Unfortunately, we don't get this for nothing because calculation of numerical values for (11.20) can involve a fair bit of work.

- We can keep track of the number of edges in a graph. Let $g_{n,q}$ be the number of simple graphs with $V = \underline{n}$ that have exactly q edges and define $c_{n,q}$ similarly. Since each of the $\binom{n}{2}$ elements of $\mathcal{P}_2(V)$ may be an edge or not, $\sum_q g_{n,q}z^q = (1+z)^{\binom{n}{2}}$. Thus

$$G(x,z) \;=\; \sum_{n\ge0}(x^n/n!)\sum_{q\ge0} g_{n,q}z^q \;=\; \sum_{n=0}^{\infty}(1+z)^{\binom{n}{2}}(x^n/n!)$$

and $C(x,z) = \ln(G(x,z))$. Unfortunately, no simple formula is known for the sum.

Special collections of graphs: We can limit our attention to particular subsets of the set of all labeled graphs. How is this done?

Let $\mathcal{C}$ be some set of connected graphs and let $G(\mathcal{C})$ be all the graphs whose connected components lie in this collection. Suppose the collection $\mathcal{C}$ satisfies the condition that the number of n vertex graphs in the set $\mathcal{C}$ depends on n and not on the labels of the vertices. In this situation, we can still derive our equation $G = e^C$ and can still keep track of other objects besides vertices. Let's look at some examples.

- Suppose that the connected components are complete graphs; i.e., every vertex in the component is connected to every other. The only thing that distinguishes one component from another is its set of vertices. Thus we can identify such a graph with a partition of its vertex set in which each block corresponds to the vertex set of a component. This is a bijection between this class of graphs and partitions of sets. We easily have $c_n = 1$ for all $n > 0$ and so $C(x) = e^x - 1$. Consequently, $G(x) = \exp(e^x - 1)$. The number of components of such a graph corresponds to the number of blocks in the partition. Thus $G(x,y) = \exp((e^x - 1)y)$, in the notation of (a), is the generating function for the Stirling numbers of the second kind. Hence Example 11.7 is a special case of our $G = e^C$ formula.

- Our next illustration, cycles of a permutation, will be a separate example. □

Example 11.11 Permutations and their cycles Let L be a set of positive integers and let s_n be the number of permutations of $\underline{n}$ such that all the cycle lengths are in L. Some examples are

(a) L is all positive integers so s_n counts all permutations;

(b) $L = \{2, 3, \ldots\}$ so s_n counts derangements;

(c) $L = \{1, 2\}$ so s_n counts involutions.

We'll obtain a generating function for s_n. In this case, the labels are simply the integers $\underline{n}$ that are being permuted.

One approach is to draw a directed graph with vertex set $\underline{n}$ and an edge (i,j) if the permutation maps i to j. This is a graph whose components are cycles and the lengths of the cycles are all in L. We can then use the approach in the previous example. This is left for you to do. We'll "forget" the previous example and go directly to the Exponential Formula.

We can construct a permutation, by choosing its cycles. Let a structure in $\mathcal{C}$ be a cycle. The parts of the structure are the things permuted by the cycle. Let a structure in $\mathcal{S}$ be a permutation. When a permutation is rooted by choosing an element of $\underline{n}$, it breaks up into a rooted cycle and an unrooted permutation. Thus the Exponential Formula can be used.

Let c_n be the number of n long cycles that can be made from $\underline{n}$. By the Exponential Formula,

$$C(x) \;=\; \mathsf{E}_{\mathcal{C}}(x) \;=\; \sum_{n\ge 1} c_n \frac{x^n}{n!}$$

and

$$S(x) \;=\; \mathsf{E}_{\mathcal{S}}(x) \;=\; e^{C(x)} \;=\; \exp\Bigl(\sum_{n\ge 1} c_n \frac{x^n}{n!}\Bigr).$$

We need to determine c_n. Since all cycle lengths must lie in L, $c_n = 0$ when $n \notin L$. Suppose $n \in L$. To construct a cycle $f\colon \underline{n} \to \underline{n}$, we specify $f(1)$, $f^2(1) = f(f(1))$, $f^3(1)$, …, $f^{n-1}(1)$. Since we want a cycle of length n, $f^n(1) = 1$ and the values $1, f(1), f^2(1), \ldots, f^{n-1}(1)$ are all distinct. Since these are the only conditions, $f(1), f^2(1), \ldots, f^{n-1}(1)$ can be any permutation of $2, 3, \ldots, n$. Thus $c_n = (n-1)!$. It follows that

$$S(x) \;=\; \exp\Bigl(\sum_{k\in L} x^k/k\Bigr). \tag{11.22}$$

Let's reexamine the three examples of L that were mentioned above. Let y stand for the sum in (11.22).

(a) When L is all positive integers,

$$y \;=\; \sum_{k=1}^{\infty} \;=\; -\ln(1-x)$$

and so $S(x) = 1/(1-x) = \sum_{n\ge 0} x^n = \sum_{n\ge 0} n!x^n/n!$. Hence $s_n = n!$, which we already knew—there are $n!$ permutations of $\underline{n}$.

(b) For derangements, y equals its value in (a) minus x and so

$$S(x) = \frac{e^{-x}}{1-x},$$

which we obtained in Example 11.4 (p. 318) by other methods.

(c) For involutions, $y = x + x^2/2$ and so

$$S(x) = \sum_{k=0}^{\infty} \frac{(x+x^2/2)^k}{k!} = \sum_{k=0}^{\infty}\sum_{j=0}^{k} \binom{k}{j} \frac{x^{k-j}(x^2/2)^j}{k!}.$$

Collecting the terms with $k + j = n$, we obtain

$$s_n = \sum_{j=0}^{n} \frac{n!}{j!\,2^j(n-2j)!},$$

which we obtained by a counting argument in Theorem 2.2 (p. 48). □

Example 11.12 Permutations and their cycles (continued) We can keep track of other information as well, provided it suffices to look at each cycle separately. In an extreme case, we could keep track of the number of cycles of each length. This requires an infinite number of classes of unlabeled parts, one for each cycle length. Let the associated variable for cycles of length k be z_k. The resulting generating function will be an EGF in the size of the set being permuted, but an ordinary generating function in each z_k since cycles are not labeled. You should be able to show that the generating function is

$$\exp\left(\sum_{k\in L} \frac{z_k x^k}{k}\right). \qquad 11.23$$

If we simply keep track of the size of the set being permuted and the number of cycles in the permutation, the generating function is just the $G(x, y)$ of (11.20). You should be able to see why this is so . If not, the second paragraph of the previous example may help. Another way to see it is to use (11.23) with $z_k = y$ for all k. Let $z(n, k)$ be the number of permutations of $\underline{n}$ that have exactly k cycles. These are called the *signless Stirling numbers of the first kind*. We have just seen that

$$Z(x, y) = \sum_{n,k\geq 0} z(n, k)(x^n/n!)y^k = \left(\frac{1}{1-x}\right)^y = (1-x)^{-y}. \qquad 11.24$$

Equating coefficients of $x^n/n!$, we obtain

$$\sum_k z(n, k)y^k = (-1)^n n! \binom{-y}{n} = y(y + 1)\cdots(y + n - 1).$$

We can use our generating function to study the expected number of cycles. The method for doing so was worked out in (10.21) (p. 284). Review that equation. Now the random variable X_n is the number of cycles and our generating function is $Z(x, y)$ instead of $A(x, y)$ and it is exponential in x. Since it is exponential, we should have a factor of $n!$ in both the numerator and denominator of (10.21) but, since it appears in both, it cancels out. On with the calculations! $Z(x, 1) = (1-x)^{-1} = \sum_{n\geq 0} x^n$, so the denominator in (10.21) is 1. We have $Z_y(x, y) = \ln(1-x)(1-x)^{-y}$ and so

$$Z_y(x, 1) = \frac{1}{1-x}(-\ln(1-x)) = \frac{1}{1-x}\sum_{k\geq 1}\frac{x^k}{k}$$

by Taylor's theorem for $-\ln(1-x)$. Finally $[x^n]\,Z_y(x, 1) = \sum_{k=1}^{n} \frac{1}{k}$, which you can work out by expanding (11.24) further or by using Exercise 10.1.6 (p. 274).

We've just worked out that the average number of cycles in a permutation of $\underline{n}$ is $\sum_{k=1}^{n} \frac{1}{k}$, a result that was derived by other means in Example 2.8 (p. 47). Using Riemann sum approximations as we did in deriving (10.30) from (10.29), it follows that the average number of cycles in a permutation of $\underline{n}$ is $\ln n + O(1)$ as $n \to \infty$.

Let's work out the variance using (10.22) (p. 285). We have $Z_{yy}(x,y) = (-\ln(1-x))^2(1-x)^{-y}$. Proceeding as in the previous paragraph,

$$[x^n]\, Z_{yy}(x,1) \;=\; \sum_{k=0}^{n} [x^k]\,(-\ln(1-x))^2 \;=\; \sum_{k=0^n} \sum_{i+j=k} \frac{1}{i}\frac{1}{j} \;=\; \sum_{i+j\le n} \frac{1}{i}\frac{1}{j}.$$

Thus

$$\begin{aligned} \mathbf{var}(X_n) \;&=\; \sum_{i+j\le n} \frac{1}{i}\frac{1}{j} + \mathbf{E}(X_n) - \left(\sum_{i=1}^{n} \frac{1}{i}\right)^2 \\ &=\; \sum_{i+j\le n} \frac{1}{i}\frac{1}{j} + \mathbf{E}(X_n) - \sum_{i,j\le n} \frac{1}{i}\frac{1}{j} \;=\; \mathbf{E}(X_n) - \sum_{\substack{i,j\le n\\ i+j>n}} \frac{1}{i}\frac{1}{j}. \end{aligned}$$

We've already done a lot of computations, so we'll just remark without proof that

$$\sum_{\substack{i,j\le n\\ i+j>n}} \frac{1}{i}\frac{1}{j} \;\sim\; \int_0^1 \frac{-\ln(1-x)}{x}\,dx \;=\; \sum_{n=1}^{\infty} \frac{1}{n^2} \;=\; \frac{\pi^2}{6}.$$

Thus $\mathbf{var}(X_n) \sim \mathbf{E}(X_n) \sim \ln n$. By Chebyshev's inequality (C.3) (p. 385), it's unlikely that $|X_n - \ln n|/(\ln n)^{1/2}$ will be large. □

***Example 11.13 Permutations and their cycles (concluded)** You should review the discussion of parity of a permutation in Definition 2.4 (p. 49) and Theorem 2.3. Let L be a set of positive integers. Let e_n (resp. o_n) be the number of even (resp. odd) permutations of $\underline{n}$ all of whose cycle lengths are in L. Let $p_n = e_n - o_n$. Using the previous example and Theorem 2.3(c), you should be able to show that the exponential generating function for p_n is given by

$$P(x) \;=\; \exp\!\left(\sum_{k\in L} \frac{(-1)^{k-1}x^k}{k}\right).$$

Let's look at some special cases.

If L be the set of all positive integers, then

$$\sum_{k\in L} \frac{(-1)^{k-1}x^k}{k} \;=\; \sum_{k=1}^{\infty} \frac{(-1)^{k-1}x^k}{k} \;=\; \ln(1+x)$$

and so $P(x) = 1 + x$. In other words, when $n > 1$, there are as many even permutations of $\underline{n}$ as there are odd and so $e_n = o_n = n!/2$. We already knew this in Theorem 2.3.

If $L = \{2, 3, \ldots\}$, we are looking at derangements. In this case

$$\sum_{k\in L} \frac{(-1)^{k-1}x^k}{k} \;=\; \sum_{k=1}^{\infty} \frac{(-1)^{k-1}x^k}{k} - x \;=\; \ln(1+x) - x$$

and so $P(x) = (1+x)e^{-x}$. With a little algebra, $p_n = (-1)^{n-1}(n-1)$. Thus the number of even and odd derangements of $\underline{n}$ differ by $n-1$, and there are more even derangements if and only if n is odd. This is a new result for us, and it's not clear how to go about proving it without generating functions.

Suppose L consists of all positive integers except $k > 2$. Reasoning as in the previous paragraph, $P(x) = (1+x)e^{\pm x^k/k}$, where the sign is plus when k is odd. If you expand $P(x)$ in a power series, you should see that $pn \ne 0$ if and only if n is a multiple of k or one more than a multiple of k. We have shown that, for $k > 2$, among the permutations of $\underline{n}$ with no k-cycles, the numbers of even and odd permutations are equal if and only if neither n nor $n-1$ is a multiple of k. □

Example 11.14 Rooted labeled trees Let's study t_n, the number of labeled rooted trees with $V = \underline{n}$.

If we remove the root vertex from a rooted tree, making all of its sons new root vertices, we obtain a graph all of whose components are rooted labeled trees. This process is reversible.

Let T be the EGF for the t_n's. By the Exponential Formula, the generating function for graphs all of whose components are rooted labeled trees is e^T. Call all such graphs $\mathcal{F}$. Thus $\mathsf{E}_{\mathcal{F}} = e^T$. The process we described in the previous paragraph constructs a rooted tree by

- partitioning the labels into (K, L) with $|K| = 1$,
- assigning the label in K to the new root,
- choosing an element $F \in \mathcal{F}$ with labels L, and
- joining the roots of the trees in F to the new root.

By the Rule of Product $T = x\mathsf{E}_{\mathcal{F}} = xe^T$.

How can we obtain t_n from the equation $T = xe^T$? There is a technique, called *Lagrange inversion*, which can be useful in this situation.

Theorem 11.6 Lagrange Inversion *Let $T(y)$, $f(y)$ and $g(y)$ be power series with $f(0) \neq 0$. Suppose that $T(x) = xf(T(x))$. Then the coefficient of x^n in $g(T(x))$ is the coefficient of u^{n-1} in $g'(u)(f(u))^n/n$; that is, $[x^n]\, g(T(x)) \;=\; [u^{n-1}]\,\big(g'(u)f(u)^n/n\big)$.*

Proofs and generalizations of Lagrange inversion are discussed at the end of this chapter.

In our particular case, $g(u) = u$ and $f(u) = e^u$. Thus

$$t_n/n! \;=\; [u^n]\, e^{nu}/n \;=\; (n^n/n!)/n.$$

Thus $t_n = n^{n-1}$.

Incidentally, we choose the symbol $\mathcal{F}$ because graphs whose components are trees are called *forests*. □

Exercises

*11.2.1. Let $\mathcal{T}$ be a collection of structures, each of which can have labeled unlabeled parts. An example is the partitions of an n-set counted by $n > 0$ and by the number of blocks. Let $T(x, y)$ be the generating function for $\mathcal{T}$, where it is exponential in the labeled parts (using x) and ordinary in the unlabeled parts (using y). State and prove a Rule of Product for these generating functions.

11.2.2. Let $\mathcal{T}$ be a collection of structures, each of which has at least one labeled part. Let $\mathsf{E}_{\mathcal{T}}$ be the EGF for $\mathcal{T}$, with respect to these labeled parts. Prove the following results and compare them with those in Exercise 10.4.1 (p. 298). When talking about lists (or sets) of structures in this exercise, the labels that are used for the objects all differ; that is, structures at different positions in a list (or set) must have different labels. If a totality of n objects in a class appear, they are to use the labels in $\underline{n}$.

(a) The EGF for k-lists of structures is $\big(\mathsf{E}_{\mathcal{T}}\big)^k$.

(b) The EGF for lists of structures is $(1 - \mathsf{E}_{\mathcal{T}})^{-1}$. Here lists of any length are allowed, including the empty list.

(c) The EGF for sets of structures, where each structure must come from $\mathcal{T}$ is $\exp\big(\mathsf{E}_{\mathcal{T}}\big)$.

(d) The EGF for circular lists is $-\ln\big(1 - \mathsf{E}_{\mathcal{T}}\big)$.

These results could be generalized to allow for unlabeled parts by using the result of Exercise 11.2.1.

11.2.3. Let $z(n,k)$ be the signless Stirling numbers of the first kind, defined in Example 11.12.

(a) Let $Z_n(y) = \sum_k z(n,k)y^k$. Show that $Z_n(y) = Z_{n-1}(y) \times (y+n-1)$ and use this to deduce the recursion

$$z(n,k) \;=\; z(n-1,k-1) + (n-1)z(n-1,k).$$

(b) Prove the previous recursion by a direct counting argument using the fact that $z(n,k)$ is the number of permutations of $\underline{n}$ with exactly k cycles.

11.2.4. Let a_n be the number of ways to place n labeled balls into boxes numbered $1, 2, \cdots$ so that the number of balls in the kth box is a multiple of k. (This is the labeled analog of the box–ball interpretation of partitions of a number.) The EGF for a can be written as a product of sums. Do it.

11.2.5. Let a_n be the number of sequences of A's B's and C's such that any letter that actually appears in the sequence must appear an odd number of times.

(a) Show that the EGF for a_n is $\left(1 + \dfrac{e^x - e^{-x}}{2}\right)^3$.

(b) Show that for n odd $a_n = (3^n + 9)/4$ and for $n > 0$ and even $a_n = 3 \times 2^{n-1}$.

11.2.6. Let $a_{n,k}$ be as in Exercise 11.2.5, except that k different letters are used instead of just A, B and C.

(a) Obtain a generating function $\sum_n a_{n,k}x^n/n!$.

(b) Show that $a_{n,k}$ equals $\sum \binom{k}{j} 2^{-j} c_{n,j}$, where the sum is over all j with the same parity as n and $\sum_n c_{n,j}x^n/n! = (e^x - e^{-x})^j$.

*(c) Can you obtain a more explicit formula for $a_{n,k}$?

11.2.7. Recall that B_n is the number of partitions of the set $\underline{n}$. Let $B(x)$ be the EGF. Show that $B'(x) = e^x B(x)$ and use this to show that

$$B_{n+1} \;=\; \sum_{k=0}^{n} \binom{n}{k} B_k.$$

11.2.8. Let a_n be the number of partitions of $\underline{n}$ such that each block has an odd number of elements and let $A(x)$ be the EGF. Use the Exponential Formula to show that $A(x) = \exp(e^x - e^{-x})/2)$.

11.2.9. Let $C(x)$ and $G(x) = \exp(C(x))$ be the EGFs for some collection of connected graphs and some collection of graphs, respectively.

(a) Let $H(x) = C(x)G(x)$ be an EGF for the sequence h_n. Using the Exponential Formula, show that the average number of components in the graphs counted by g_n is h_n/g_n.

(b) Prove the formula in the previous part by a simple application of the Rule of Product.
Hint. Remember that $H(x)$ will be counting the *total* number of components, not the average number.

(c) Deduce the formula at the end of Example 11.12 for the average number of cycles in a permutation.

(d) Deduce that the average number of blocks in a partition of $\underline{n}$ is $\frac{B_{n+1}}{B_n} - 1$.

(e) Obtain a formula like the previous one for the average number of cycles in an involution.

11.2.10. Let a_n be the number of partitions of $\underline{n}$ where the order of the blocks is important. Obtain the EGF $A(x)$ and use it to show that $a_n = \sum_{k>0} k^n/2^{k+1}$.

11.2.11. A permutation f of $\underline{n}$ is said to be *alternating* if $f(1) < f(2) > f(3) < \ldots$. Let a_n be the number of alternating permutations of $\underline{n}$. It will be convenient to set $a_0 = 1$. Let A be the EGF for the a_n's and let B be the EGF for those a_n with odd n; that is, $b_{2n} = 0$ and $b_{2n+1} = a_{2n+1}$. By considering $f(1), \ldots, f(k-1)$ and $f(k+1), \ldots, f(n)$, where $k = f^{-1}(n)$, show that

$$B'(x) \;=\; B(x)^2 + 1 \quad\text{and}\quad A'(x) = B(x)A(x) + 1.$$

Verify that $A(x) = \tan x + \sec x$ and $B(x) = \tan x$.

11.2.12. A k-ary tree is a rooted tree in which each nonleaf vertex has exactly k sons. Let t_n be the number of unlabeled plane k-ary trees having n nonleaf vertices.

(a) Prove that the ordinary generating function for t_n satisfies the equation $T(x) = 1 + xT(x)^k$.

(b) Use Lagrange inversion (Theorem 11.6) to show that $t_n = \frac{1}{n}\binom{nk}{n-1}$.
Hint. Apply the theorem to $S(x) = T(x) - 1$.

11.2.13. A function $f\colon A \to A$ is said to have a square root if there is a function $g\colon A \to A$ such that $g(g(a)) = f(a)$ for all $a \in A$.

(a) Show that a permutation has a square root if and only if, it has an even number of cycles of length $2k$ for each $k > 0$.

(b) Let s_n be the number of permutations of $\underline{n}$ which have a square root. Show that

$$S(x) = \sum_{n\geq 0} s_n x^n/n! = \exp\left(\sum_{\substack{k=1\\k \text{ odd}}}^{\infty} \frac{x^k}{k}\right) \prod_{\substack{k=2\\k \text{ even}}}^{\infty} \frac{\exp(x^k/k) + \exp(-x^k/k)}{2}.$$

Hint. One way is to use $\exp\left(\sum_{k=1}^{\infty} z_k x^k/k\right)$ and then use bisection of series for each even k one by one. Another approach is to do each cycle length separately and then use the Rule of Product.

(c) Using $\cosh(u) = (e^u + e^{-u})/2$ and bisection of the Taylor series for $\ln(1-x)$, show that

$$S(x) = \sqrt{\frac{1+x}{1-x}} \prod \cosh(x^{2k}/2k).$$

(d) By taking logarithms, differentiating and then multiplying by $S(x)$ conclude that $S'(x) = S(x)B(x)$ where $B(x)$ is $(1-x^2)^{-1}$ plus the sum of $x^{n-1}\tanh(x^n/n)$ over all even positive n. How much work is involved in using this to construct tables of s_n? Can you think of an easier method?

11.2.14. Let $a_{n,k}$ be the number of permutations of $\underline{n}$ having exactly k cycles of even length.

(a) Show that

$$\sum_{n,k} \frac{a_{n,k} x^n y^k}{n!} = (1-x^2)^{-y/2} \sqrt{\frac{1+x}{1-x}}.$$

(b) Conclude that the average number of even length cycles in a permutation of $\underline{n}$ is

$$\sum_{k=1}^{\lfloor n/2 \rfloor} \frac{1}{2k},$$

where the floor function $\lfloor x \rfloor$ is the largest integer not exceeding x.

(c) Show that the number of odd length cycles minus the number of even length cycles averaged over all permutations of $\underline{n}$ is

$$\sum_{k=\lfloor n/2 \rfloor + 1}^{n} \frac{1}{k}$$

and that this sum approaches $\ln 2$ as $n \to \infty$.

*11.2.15. Let $t_{n,k}$ be the number of n-vertex rooted labeled trees with k leaves. Let $L(x,y) = \sum_{n,k} t_{n,k}(x^n/n!)y^k$. (The generating function for leaves is ordinary because the labels on the leaves have already been taken into account when we counted vertices.) Let $T(x)$ be the EGF for rooted labeled trees by number of vertices.

(a) Show that $L = xe^L - x + xy$.

(b) Let $U(x)$ be the EGF $\left.\frac{\partial L(x,y)}{\partial y}\right|_{y=1}$. Show that the average number of leaves in an n-vertex rooted labeled tree is u_n/n^{n-1}.

*(c) Show that $U(x) = x^2T'(x) + x$ and use this to show that the average number of leaves in an n-vertex rooted labeled tree is $n(1-1/n)^{n-1}$. Conclude that the probability that a randomly chosen vertex is a leaf approaches $1/e$ as $n \to \infty$.

*11.2.16. In this exercise, you'll study the average height of vertices in rooted labeled trees. The height of a vertex is the number of edges on the path from it to the root. For a rooted tree T, let $h(T)$ be the sum of the heights of the vertices of T. Let $t_{n,k}$ be the number of n-vertex rooted labeled trees T with $h(T) = k$ and let $H(x,y) = \sum_{n,k} t_{n,k}(x^n/n!)y^k$. If T has n vertices, the average height of a vertex in T is $h(T)/n$. Let $\mu(n)$ be the average of $h(T)/n$ over all n-vertex rooted labeled trees.

(a) Show that

$$\mu(n) = n^{-n}\sum_k k t_{n,k}$$

and that $n^n\mu(n)$ is the coefficient of $x^n/n!$ in $D(x) = \left.\frac{\partial H(x,y)}{\partial y}\right|_{y=1}$.

(b) Show that $H(x,y) = x\exp\big(H(xy,y)\big)$.

(c) Show that

$$D(x) = \frac{xT'(x)T(x)}{1-T(x)} = \left(\frac{T(x)}{1-T(x)}\right)^2,$$

where $T(x)$ is the EGF for rooted labeled tree by vertices.

(d) Use Lagrange inversion with $g(u) = \left(\frac{u}{1-u}\right)^2$ to show that

$$\mu(n) = \frac{2\,(n-1)!}{n^n}\sum_{k=0}^{n-2}\binom{k+2}{2}\frac{n^{n-k-2}}{(n-k-2)!}.$$

One can obtain estimates of $\mu(n)$ for large n from this formula, but we will not do so.

11.2.17. A *functional digraph* is a simple digraph in which each vertex has outdegree 1. It is connected if the associated graph is connected. Let $\mathcal{F}_n$ be the set of connected n-vertex functional digraphs and let $f_n = |\mathcal{F}_n|$.

(a) Define a function φ from $\underline{n}^{\underline{n}}$ to digraphs with $V = \underline{n}$ as follows:

$$\text{for } g \in \underline{n}^{\underline{n}},\ \ \varphi(g) \ = \ (V,E) \ \text{ where } \ E \ = \ \{(x,g(x)) \mid x \in \underline{n}\}.$$

Prove that φ is a bijection from $\underline{n}^{\underline{n}}$ to the set of all functional digraphs with vertex set $\underline{n}$. (See Example 5.9 (p. 142).)

(b) Show that a connected functional digraph consists of a circular list of rooted trees, with each tree's edges directed toward the root and with the roots joined in a cycle (as indicated by the circular list). We're not asking for a proof, just a reasonable explanation of why this is true. Drawing some pictures may help.

(c) Show that $\sum_n f_n x^n/n! = -\ln(1-T(x))$, where $T(x)$ is the EGF for rooted labeled trees.

(d) Using the fact that $T(x) = xe^{T(x)}$ and Lagrange inversion, deduce

$$f_n \ = \ (n-1)!\sum_{k=0}^{n-1}\frac{n^k}{k!}.$$

11.2.18. A *rooted map* is an unlabeled graph that has been embedded in the plane and has had one edge distinguished by assigning a direction to it and selecting a side of it. Tutte developed some clever techniques for counting such structures. Using them it can be shown that

$$M(x) \;=\; (1-4u)(1-3u)^{-2} \quad \text{where} \quad x = u(1-3u)$$

and M is the ordinary generating function for m_n, the number of n-edge rooted maps. Prove that

$$m_n \;=\; \frac{2(2n)!\,3^n}{n!\,(n+2)!}.$$

Hint. The notation may be a bit confusing for using Theorem 11.6. Note that $T(x)$ is simply the function that is given implicitly, so in this case $T(x) = u$.

11.3 Symmetries and Pólya's Theorem

In this section we will discuss a generalization of the Burnside Lemma. We will then consider an important special case of this generalization, namely Pólya's Theorem. You should review the statement and proof of the Burnside Lemma (Theorem 4.5 (p. 112)).

Let S be a set with a permutation group G. Recall that we say $x, y \in S$ are equivalent if $y = g(x)$ for some $g \in G$. (These equivalence classes are referred to as *orbits* of G in S.) Suppose further that there is a function W defined on S such that W is constant on equivalence classes. This means that if x and y are equivalent, then $W(y) = W(x)$. We can rephrase "W is constant on equivalence classes" as "$W(g(x)) = W(x)$ for all $g \in G$ and all $x \in S$."

You may have noticed that W is not completely specified because we haven't defined its range. We don't really care what the range is as long as addition of range elements and multiplication of them by rationals is possible. Thus the range might be the real numbers, polynomials with rational coefficients, or lots of other things.

Let $\mathcal{E}$ be the set of equivalence classes of S with respect to the group G. (Don't confuse $\mathcal{E}$ with our notation for exponential generating functions.) If $B \in \mathcal{E}$, define $W(B) = W(y)$, where y is any element of B. This definition makes sense because W is constant on equivalence classes.

Theorem 11.7 The Weighted Burnside Lemma *With the above definitions,*

$$\sum_{B\in\mathcal{E}} W(B) \;=\; \frac{1}{|G|}\sum_{g\in G} N(g),$$

where $N(g)$ is the sum of $W(x)$ over all $x \in S$ such that $g(x) = x$.

Before reading further, you should be able to see that the case $W(x) = 1$ for all $x \in S$ is just Burnside's Lemma. This is simply a matter of understanding the notation we've introduced.

Proof: The proof of this result is a simple modification of the proof of Burnside's Lemma. Here it is. You should be able to supply the reasons for all the steps by referring back to the proof of Burnside's Lemma.

$$\begin{aligned}
\sum_{B\in\mathcal{E}} W(B) &= \sum_{B\in\mathcal{E}}\sum_{y\in B}\frac{W(B)}{|B|} = \sum_{B\in\mathcal{E}}\sum_{y\in B}\frac{W(y)}{|B_y|}\\
&= \sum_{y\in S}\frac{|I_y|}{|G|}W(y) = \frac{1}{|G|}\sum_{y\in S}\left(\sum_{g\in G}\chi(g(y)=y)W(y)\right)\\
&= \frac{1}{|G|}\sum_{g\in G}\left(\sum_{y\in S}\chi(g(y)=y)W(y)\right)\\
&= \frac{1}{|G|}\sum_{g\in G}N(g). \quad \square
\end{aligned}$$

Example 11.15 The Ferris wheel Let's return to the "Ferris wheel" problem of Example 1.12 (p. 13). Recall that we want to look at circular arrangements of ones and twos where the circles contain six digits. In this case, the group G contains 6 elements, which can be described by how they rearrange the six positions on the Ferris wheel. We called the group elements g_0 through g_5, where g_i shifts things circularly i positions.

As already noted, if we set $W(x)=1$ for all x, then we simply end up counting all equivalence classes. Suppose, instead, that we set $W(x)=1$ if x contains exactly 4 ones and $W(x)=0$ otherwise. This is an acceptable definition of W because two equivalent sequences contain the same number of ones. In this case, we end up counting the equivalence classes of sequences that contain exactly 4 ones.

A more interesting example is obtained by letting z be a variable and setting $W(x)=z^k$, where k is the number of ones in x. In this case the coefficient of z^k in $W(\mathcal{E})=\sum_{B\in\mathcal{E}}W(B)$ is the number of equivalence classes whose sequences each contain exactly k ones. In other words, $W(\mathcal{E})$ is a generating function for equivalence classes of sequences by number of ones. Let's compute $W(\mathcal{E})$ in this case. For $N(g_0)$, any sequence is allowed since $g_0(x)=x$ for all x. Thus each position can be either a two or a one. By the Rules of Sum and Product for generating functions, $N(g_0)=(1+z)^6$. If $g_1(x)=x$, all positions must be the same and so $N(g_1)=1+z^6$. If $g_2(x)=x$, the even numbered positions have the same value and the odd numbered positions must have the same value. By the Rules of Sum and Product for generating functions, $N(g_2)=(1+z^3)^2$. Similarly, $N(g_3)=(1+z^2)^3$, $N(g_4)=N(g_2)$ and $N(g_5)=N(g_1)$. Thus we obtain

$$\begin{aligned}
W(\mathcal{E}) &= \frac{1}{6}\Big((1+z)^6+2(1+z^6)+2(1+z^3)^2+(1+z^2)^3\Big)\\
&= 1+z+3z^2+4z^3+3z^4+z^5+z^6.
\end{aligned} \qquad 11.25$$

One does not need this machinery to compute the generating function. It is quite simple to construct it from the leaves of the decision tree in Figure 4.2. $\square$

We now describe a particularly important case of the Weighted Burnside Lemma. Let A and B be finite sets, let S be the functions B^A and let G be a permutation group on A. We make G into a permutation group on B^A by defining $g(f)$ for every $g\in G$ and every $f\in B^A$ to be the function given by

$$(g(f))(a)=f(g(a)) \quad \text{for all } a\in A.$$

Let W be a function from B to a set in which we can divide by integers, add and multiply. Among such sets are the set of real numbers, the set of polynomials in any number of variables, and the set of power series. (There will be a concrete example soon.) We make W into a weight function on B^A by setting

$$W(f) = \prod_{a \in A} W(f(a)).$$

Why is W constant on equivalence classes? Since f and $g(f)$ are equivalent, the second line of $g(f)$ in two line form is simply a permutation of the second line of f; however, $W(f)$ is simply the product of the weights of the elements in its second line, without regard to their order.

Example 11.16 The Ferris wheel revisited Let's return to the previous example. We can phrase it in our new terminology:

$$\begin{aligned} A &= \{1,2,3,4,5,6\}; \\ B &= \{1,2\}; \\ G &= \text{the circular shifts of 1,2,3,4,5,6;} \\ W &= \left(\begin{smallmatrix} 1 & 2 \\ z & 1 \end{smallmatrix}\right). \end{aligned}$$

For example, if $g = (1,3,5)(2,4,6)$ and $f = \left(\begin{smallmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 2 & 2 & 1 & 2 & 1 \end{smallmatrix}\right)$, then $g(f) = \left(\begin{smallmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 2 & 1 & 1 & 1 & 2 \end{smallmatrix}\right)$ and $W(f) = z^3$. You should be able to verify the following observations.

(a) An element of B^A can be viewed as a 6-long sequence of ones and twos.

(b) G permutes these 6-long sequences just as the G in the previous example did.

(c) $W(f)$ is z raised to a power which equals the number of ones in the sequence $f(1), \ldots, f(6)$.

These observations show that this problem is the same as the previous one. $\square$

Why is this special situation with $S = B^A$ important? First, because many problems that are done with the Weighted Burnside Lemma can be phrased this way. Second, because it is easier to apply the lemma in this particular case. The method for applying the lemma is known as Pólya's Theorem. Before stating the theorem, we'll look at a special case.

Example 11.17 The Ferris wheel revisited In order to compute $N(g)$ for the Ferris wheel, we need to study those functions f such that $g(f) = f$. Look at $g = (1,3,5)(2,4,6)$ again. You should be able to see that $(g(f))(a) = f(a)$ for all $a \in A$ if and only $f(1) = f(3) = f(5)$ and $f(2) = f(4) = f(6)$. For example $(g(f))(1) = f(g(1)) = f(3)$, and so $g(f) = f$ implies that $f(3) = f(1)$. More generally, you should be able to see that for any permutation g, we have $g(f) = f$ if and only if f is constant on each of the cycles of g.

To compute the sum of the weights of the functions f with $g(f) = f$, we can look at how to construct such a function:

1. First choose a value for f on the cycle $(1,3,5)$ AND
2. then choose a value for f on the cycle $(2,4,6)$.

On the first cycle, the value of f is either one OR two. If f is one, this part of f contributes z^3 to the weight $W(f)$. If f is two, this part of f contributes 1 to the weight $W(f)$. Using the Rules of Sum and Product, we get that all f with $g(f) = f$ contribute a total of $(z^3+1)(z^3+1)$; that is, $N(g) = (z^3+1)^2$. $\square$

In order to state Pólya's Theorem, we must define the *cycle index* Z_G of a group G of permutations. Let $c_i(g)$ be the number of cycles of g of length i. Then

$$Z_G(x_1, x_2, \ldots) = \frac{1}{|G|} \sum_{g \in G} {x_1}^{c_1(g)} {x_2}^{c_2(g)} \cdots .$$

If G is the cyclic shifts of the sequence $1, 2, 3, 4, 5, 6$, you should be able to show that

$$Z_G = \frac{1}{6}\Big({x_1}^6 + 2x_6 + 2{x_3}^2 + {x_2}^3\Big). \qquad 11.26$$

Theorem 11.8 Pólya's Theorem *If $S = B^A$ and G and W are defined as above, then*

$$\sum_{E \in \mathcal{E}} W(E) = Z_G(x_1, x_2, \ldots),$$

where

$$x_i = \sum_{b \in B} (W(b))^i.$$

This can be proved using the idea that we introduced in the previous example together with the Weighted Burnside Lemma. The proof is left as an exercise. You will probably find it easier to prove after reading some examples.

Example 11.18 The Ferris wheel revisited Now we'll apply Pólya's Theorem to derive (11.25). Since $B = \{1, 2\}$ and $W(B) = \binom{1\ 2}{z\ 1}$, we have $x_1 = z + 1$, $x_2 = z^2 + 1$ and so on. Substituting these values into (11.26), we obtain (11.25).

Now let's consider a Ferris wheel of arbitrary length n. Our group consists of the n cyclic shifts $g_0, \ldots, g_{n-1}$, where g_i shifts the sequence circularly i positions. This group is known as the *cyclic group* of order n and is usually denoted C_n. In order to apply Pólya's Theorem, we need to compute Z_{C_n}, which we now do.

The element g_i shifts something in position p to position $p + i$, then to $p + 2i$ and so on, where all these values are reduced modulo n, which means we divide them by n and keep the remainder. For example, if $n = 6$, $i = 4$ and $p = 3$, the successive values of the positions are 3, 1 (the remainder of 7/6), 5 and back to 3. You should be able to see easily that the length of the cycle containing p depends only on n and g_i and not on the choice of p. Thus all cycles of g_i have the same length. What is that length?

Suppose we return to position p after k steps. This can happen if and only if dividing $p + ki$ by n gives a remainder of p. In other words, ki must be a multiple of n. Since ki is also a multiple of i, it must be a multiple of the least common multiple of i and n, which is written $\operatorname{lcm}(i, n)$. Thus, the smallest possible value of ki is $\operatorname{lcm}(i, n)$. It follows that

$$\text{the length of each cycle of } g_i \text{ is } \frac{\operatorname{lcm}(i, n)}{i}. \qquad 11.27$$

Since the cycles must contain n items between all of them, the number of cycles is

$$\frac{n}{\operatorname{lcm}(i, n)/i} = \frac{ni}{\operatorname{lcm}(i, n)} = \gcd(i, n),$$

where the last equality is a fairly easy number theory result (which is left as an exercise). Incidentally, this number theory result enables us to rewrite (11.27) as

$$\text{the length of each cycle of } g_i \text{ is } \frac{n}{\gcd(i, n)}.$$

It follows from all this discussion, that g_i contributes the term $(x_{n/k})^k$, where $k = \gcd(n,i)$, to the sum for the cycle index of C_n. Thus

$$Z_{C_n} \;=\; \frac{1}{n}\sum_{i=0}^{n-1}\bigl(x_{n/\gcd(n,i)}\bigr)^{\gcd(n,i)}. \qquad 11.28$$

With $n = 6$, (11.28) gives us (11.26), as it should. Carry out the calculation. Notice that some of the terms were equal so we were able to collect them together. It would be nice to do that in general. This means we need to determine when various values of $\gcd(n,i)$ occur. We leave this as an exercise. □

Example 11.19 Unlabeled rooted trees Although we have studied the number of various types of unlabeled RP-trees, we haven't counted those that are not in the plane. There's a good reason for that—we need Pólya's theorem or something similar.

We'll look at unlabeled rooted trees where each vertex has at most three edges directed away from the root. Let t_n be the number with n vertices. We want to study $T(x)$, the ordinary generating function for the sequence $t_0, t_1, \ldots$. There are various reasons for doing this. First, it is not too difficult and not too easy. Second, these trees correspond to an important class of organic compounds: Each of the vertices corresponds to a carbon atom. These carbon atoms are all joined by single bonds. A "radical" (for example, –OH or –COOH) is attached by a single bond to the carbon atom that corresponds to the root Finally, to give each carbon atom valency four, hydrogen atoms are attached as needed. Compounds like this with the –OH radical are known as *alcohols*. Two alcohols with the same number of carbon atoms but different associated trees are called *isomers*. The two isomers of propyl alcohol are

$$\begin{array}{ccccccc} & H & & OH & & H & \\ & | & & | & & | & \\ H- & C & - & C & - & C & -H \\ & | & & | & & | & \\ & H & & H & & H & \end{array} \qquad\qquad \begin{array}{ccccccc} & H & & H & & H & \\ & | & & | & & | & \\ H- & C & - & C & - & C & -OH \\ & | & & | & & | & \\ & H & & H & & H & \end{array}$$

The corresponding rooted trees are $\circ\!-\!\bullet\!-\!\circ$ and $\circ\!-\!\circ\!-\!\bullet$, respectively, where $\bullet$ indicates a root and $\circ$ a nonroot.

We can approach this problem the same way we did RP-trees: We take a collection of unlabeled rooted trees and join them to a new root. There are two important differences from our previous considerations of RP-trees.

- Since a vertex has at most three sons, we must take this into account. (Previously we dealt mostly with exactly two sons.)
- There is *no ordering among the sons*. This is what we mean by not being in the plane—the sons are not ordered from left to right; they are simply a multiset of trees. In terms of symmetries, this means that all permutations of the sons give equivalent trees.

Let's begin with the problem of at most three sons. One way to handle this is to sum up the cases of exactly one, two and three sons. There is an easier way. We will allow the empty tree, so $t_0 = 1$. By taking three trees, we get at most three sons since any or all of them could be the empty tree.

Pólya's Theorem can be applied in this situation. In the notation of the theorem, $A = \underline{3}$ and B is the set of rooted trees that we are counting. A function in B^A selects a list of three trees to be the sons of the root. Since all permutations of these sons are possible, we want to study the group of all possible permutations on three things. This group is known as the symmetric group on three things and is denoted S_3. More generally, one can speak of S_n. Since all permutations of n things are allowed, S_n contains $n!$ elements, the number of permutations of an n-set.

By writing down all six permutations of $\underline{3}$, you should be able to easily show that

$$Z_{S_3} \;=\; \frac{x_1^3 + 3x_1x_2 + 2x_3}{6}.$$

We need to compute x_i so that we can apply Pólya's Theorem. As noted earlier, B is the set of all unlabeled rooted trees of the sort we are constructing. $W(b)$ for a tree b is simply x^k, where k is the number of vertices of b. It follows that $x_i = T(x^i)$. Thus we have

$$T(x) \;=\; 1 + x\frac{T(x)^3 + 3T(x)T(x^2) + 2T(x^3)}{6}, \qquad 11.29$$

where the "1+" is present because a (possibly empty) tree is constructed by taking the empty set OR ....

Equation (11.29) can be used to compute t_n recursively. You should be able to do this. If you do this for n up to five or so, you will discover that it is probably easier to simply list the trees and then count them. On the other hand, suppose you want the answer up to $n = 20$. You will probably want to use a computer. While it is certainly possible to write a program to list the trees, it is probably easier to use a symbolic manipulation package. Simply start with $T(x) = 1$, which is obviously the beginning of the generating function. Then apply (11.29) n times to compute new values of $T(x)$. To avoid overflow and/or excessive running time, you should truncate all calculations to terms of degree at most n.

There is another situation in which (11.29) is better than listing. Suppose that we want to get an estimate of, say t_{100}. Asymptotic methods provide a way for doing this using (11.29). See Example 11.34 (p. 354). □

In general, computing the cycle index of a group is not a simple matter. The examples considered so far have involved relatively simple groups. The following example deals with a somewhat more complicated situation—the cycle index of the group of symmetries of a cube, where the group is a permutation group on the faces of the cube.

Example 11.20 Symmetries of the cube In Exercise 4.2.7 (p. 110), you used a decision tree to study the ways to color the faces of a cube. Here we'll use a cycle index polynomial to study it. Before doing this, we must answer two questions:

- What symmetries are possible? Certainly, we should allow the rotations of the cube. There are other symmetries that involve reflections. Whether we allow these or not will depend on the problem at hand. A solid cube in the real world can only be rotated; however, reflections of a cube that is associated with a chemical compound (like the trees in the previous example) may be a perfectly acceptable physical manipulation. We'll begin with just rotations and then allow reflections as well.
- What objects are being permuted? Obvious choices are the vertices, edges and faces of the cube. There are less obvious ones such as diagonals. Different choices for the objects will, in general, lead to different permutation groups and hence different cycle index polynomials. Since we are coloring faces, we'll choose the faces of the cube and leave other objects as exercises.

Before proceeding, we recommend that you find a cube if you can. Possibilities include a sugar cube, a die and a homemade cube consisting of six squares of cardboard taped together. Here is a picture of an "unfolded" cube.

$$\begin{array}{|c|c|c|c|} \cline{2-2} \multicolumn{1}{c|}{} & 1 & \multicolumn{2}{c}{} \\ \hline 2 & 3 & 4 & 5 \\ \hline \multicolumn{1}{c|}{} & 6 & \multicolumn{2}{c}{} \\ \cline{2-2} \end{array} \qquad 11.30$$

If you imagine the square marked 3 as being the base, squares 1, 2, 4 and 6 fold up to produce sides and square 5 folds over to become the top.

What axes can the cube be rotated around and through what angles so that it will occupy the same space? The axes fall into three classes:

- Face centered: This type goes through the centers of opposite pairs of faces. There are three of them, through the faces 1-6, 2-4 and 3-5. A cube can be rotated 0°, ±90° or 180° about such an axis.

- Edge centered: This type goes through the centers of opposite pairs of edges. There are six of them. An edge can be described by the two faces that lie on either side of it. In this notation, one edge centered axis is 25-34. A cube can be rotated 0° or 180° about such an axis.
- Vertex centered: This type goes through opposite vertices of the cube. There are four of them, one of which is 125-346, where a vertex is described by the three faces that meet there. A cube can be rotated 0° or ±120° about such an axis.

To compute a term in the cycle index polynomial corresponding to a rotation, we can look at how the rotation permutes the faces and then determine the term from the cycle lengths. For example, the face centered rotation about 1-6 through 90° gives the permutation $(1)(2,3,4,5)(6)$. This gives us the term $x_1x_4x_1 = x_1^2x_4$. You should be able to establish the following terms by studying your cube. The letters F, E and V describe the type of axis.

$$\begin{array}{rl} \text{no rotation} & x_1^6 \\ \pm 90^\circ \text{ F} & x_1^2x_4 \\ 180^\circ \text{ F} & x_1^2x_2^2 \\ 180^\circ \text{ E} & x_2^3 \\ \pm 120^\circ \text{ V} & x_3^2 \end{array}$$

Altogether there are 24 rotations. (We have counted the rotations through 0° just once.)

Before we add up the 24 terms, we must verify that the rotations are all distinct. One way to do this is by looking at the cycles of the 24 permutations of the faces and noting that they are distinct. Can you think of another method for seeing that they are distinct? You might try a different approach—instead of showing that the rotations are distinct directly, show that there must be 24 distinct rotations by a geometric argument and then use the fact that we have found all possible rotations.

Adding up the 24 terms, the cycle index polynomial for the rotations of the cube in terms of the faces of the cube is

$$\frac{x_1^6 + 6x_1^2x_4 + 3x_1^2x_2^2 + 6x_2^3 + 8x_3^2}{24}. \qquad 11.31$$

It follows that the number of rotationally inequivalent ways to color the faces of a cube using k colors is

$$C(k) = \frac{k^6 + 6k^3 + 3k^4 + 6k^3 + 8k^2}{24} = \frac{k^6 + 3k^4 + 12k^3 + 8k^2}{24}. \qquad 11.32$$

How many rotationally inequivalent ways can the cube be colored with 3 colors so that every color appears? We cannot substitute directly into the cycle index polynomial, but we can see the answer with a little thought. Can you do it?

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

One solution is to use the Principle of Inclusion and Exclusion together with (11.32). The answer is

$$C(3) - 3C(2) + 3C(1) - C(0) = 57 - 3 \times 10 + 3 \times 1 - 0 = 30.$$

How many rotationally inequivalent ways can we color the faces of a cube with k colors so that adjacent faces have distinct colors? This problem cannot be answered with Pólya's Theorem; however, it can be done with Burnside's Lemma.

We now turn our attention to rotations and reflections of the cube. Imagine the cube made from (11.30) has been rotated in any fashion. Now carry out the following operations.

- Rotate it so that face 3 is on the bottom and face 2 is on the left.
- Open the cube by cutting two sides of face 4 and all sides of face 5 except the one between it and face 4.

This will always lead to the picture in (11.30). Now suppose you do the same thing with a cube that has been rotated and, possibly, reflected. The result will be either (11.30) or (11.30) with 1 and 6 interchanged. You should convince yourself of this by experimentation or by geometric arguments.

The result of the previous paragraph implies that any reflection and rotation combination can be obtained as either a rotation or a rotation followed by an interchange of the labels 1 and 6. Using this observation, the cycle index of the group of rotations and reflections can be computed from a knowledge of the orbits of the rotations. We will not carry out the tedious details. The result is

$$\frac{x_1^6 + 3x_1^4x_2 + 6x_1^2x_4 + 9x_1^2x_2^2 + 7x_2^3 + 6x_2x_4 + 8x_3^2 + 8x_6}{48}$$

There are alternative geometric arguments that could be used to obtain this result. For example, one can look at the possible arrangement of faces around the upper left front corner of the cube. □

***Example 11.21 Counting unlabeled graphs** How many unlabeled graphs are there with n vertices and q edges? This has many variants: do we allow multiple edges? loops? Are the graphs directed?

All of these can be done in a similar manner and all lead to messy expressions. We'll choose the simplest case: simple directed graphs with loops allowed.

In any of these situations, we use Pólya's Theorem. Suppose that n, the number of vertices, is given. Let $\underline{n}$ be the set of vertices. The functions we consider will be from $\underline{n} \times \underline{n}$ to $\{0, 1\}$. The value of $f((u, v))$ is the number of edges from u to v. In the notation of Pólya's Theorem, $S = B^A$ and G acts on A. We have already said that $B = \{0, 1\}$ and $A = \underline{n} \times \underline{n}$. What is G and what is the weight W? The group G will be the group of all permutations of n things, but instead of acting on the vertices $\underline{n}$, it must act on the ordered pairs $\underline{n} \times \underline{n}$. Most of this example will be devoted to explaining and computing the cycle index of this group action. W is given by $W(i) = y^i$. The coefficient of y^q will then be the number of unlabeled simple digraphs with n vertices and q edges.

Before turning to the calculations, we remark how some of the other graph counting problems can be dealt with. If loops are not allowed, remove the n ordered pairs of the form (i, i) from A. If any number of edges is allowed, replace B by the nonnegative integers, still setting $W(i) = y^i$. To count (loopless) graphs, replace $\underline{n} \times \underline{n}$ with $\mathcal{P}_2(\underline{n})$, the set of 2 element subsets of $\underline{n}$.

Let g be a permutation acting on $\underline{n}$. If we write g in cycle form, it is fairly easy to translate g into a permutation of $\underline{n} \times \underline{n}$. For example, suppose that $n = 3$ and $g = (1, 2)(3)$. To avoid confusing ordered pairs with cycles, we will indicate an ordered pair without parentheses and commas, e.g., 13 instead of $(1, 3)$. Using g, we have the following two line form for the corresponding permutation of $\underline{3} \times \underline{3}$

$$\begin{pmatrix} 11 & 12 & 13 & 21 & 22 & 23 & 31 & 32 & 33 \\ 22 & 21 & 23 & 12 & 11 & 13 & 32 & 31 & 33 \end{pmatrix},$$

which you should be able to verify easily. In cycle form this is

$$(11, 22)(12, 21)(13, 23)(31, 32)(33),$$

which contributes $x_1x_2^4$ to the cycle index. How do we do this in general?

Suppose $u, v \in \underline{n}$ are two vertices, that u belongs to a cycle of g of length i and that v belongs to a cycle of length j. The length of the cycle containing uv is the number of times we must apply g in order to return to uv. After we apply g to uv k times, we will have advanced k positions in the cycle containing u and k positions in the cycle containing v. Thus, we will return to uv after k times if and only if k is a multiple of i and k is a multiple of j. The smallest positive such k is $\mathrm{lcm}(i, j)$, the least common multiple of i and j. Thus uv belongs to a cycle of length $\mathrm{lcm}(i, j)$ and this cycle contributes a factor of $x_{\mathrm{lcm}(i,j)}$ to a term of the cycle index.

Let's look more closely at the set of directed edges st that can be formed by choosing s from the same cycle as u and t from the same cycle as y. There are ij such edges. Like uv, each edge st lies in a cycle of length $\mathrm{lcm}(i, j)$. Thus, there are $ij/\mathrm{lcm}(i, j) = \gcd(i, j)$ such cycles.

Let's look carefully at what we have shown. If we choose a cycle C of length i and another cycle D of length j, then the ij ordered pairs in $C \times D$ lie in $\gcd(i,j)$ cycles of length $\mathrm{lcm}(i,j)$. Thus they contribute a factor of $x_{\mathrm{lcm}(i,j)}^{\gcd(i,j)}$ to a term of the cycle index.

If g acting on $\underline{n}$ has exactly ν_k cycles of length k, the argument in the previous paragraph shows that it contributes the term

$$\prod_i \prod_j \Big(\big(x_{\mathrm{lcm}(i,j)}\big)^{\gcd(i,j)} \Big)^{\nu_i \nu_j}$$

to the cycle index we are computing. This gives us the following recipe for computing the cycle index.

Theorem 11.9 *To compute the cycle index for the n-vertex unlabeled digraphs (with loops allowed), start with the cycle index for the set of all $n!$ permutations of $\underline{n}$. Replace every term*

$$c_i \prod_{i=1}^{n} x_i^{\nu_i} \quad \textit{with} \quad c_i \prod_{i,j} \big(x_{\mathrm{lcm}(i,j)}\big)^{\nu_i \nu_j \gcd(i,j)},$$

where the latter product extends over all (i,j). □

Exercises

11.3.1. This deals with the cycle index of C_n, the cyclic group. You will need to know that, up to the order of the factors, every number can be factored uniquely as a product of primes. We take that as given.

(a) Suppose that $A = p_1^{a_1} \cdots p_k^{a_k}$ where $p_1, \ldots, p_k$ are distinct primes. We use the shorthand notation $A = \mathbf{p}^{\mathbf{a}}$ for this product. Suppose that $B = \mathbf{p}^{\mathbf{b}}$. Let $c_i = \min(a_i, b_i)$, $d_i = \max(a_i, b_i)$, $C = \mathbf{p}^{\mathbf{c}}$ and $D = \mathbf{p}^{\mathbf{d}}$. Prove that $AB = CD$, $C = \gcd(A,B)$ and $D = \mathrm{lcm}(A,B)$. This establishes the claims in Example 11.18.

(b) Prove that the number of integers i in $\{1, \ldots, n\}$ for which $\gcd(n,i) = k$ is

- zero if k does not divide n;
- the number of integers j in $\{1, \ldots, \frac{n}{k}\}$ for which $\gcd(j, n/k) = 1$ if k divides n. This latter number is denoted $\varphi(n/k)$ and is called the *Euler phi function*. We discussed how to compute it in Exercise 4.1.5 (p. 100).

(c) Conclude that

$$Z_{C_n} = \frac{1}{n} \sum \varphi(n/k) z_{n/k}^{k} = \frac{1}{n} \sum \varphi(k) z_k^{n/k},$$

where the sum ranges over all integers between 1 and n inclusive that divide n.

11.3.2. The following questions refer to the group of rotations of the cube.

(a) Compute the cycle index of the group acting on the edges of the cube.

(b) Compute the cycle index of the group acting on the vertices of the cube.

(c) Imagine three perpendicular axes drawn through the center of the cube joining the centers of faces. Label the axes x, y and z, but *do not distinguish a direction on the axes*—thus the axes are simply lines. Compute the cycle index of the group acting on these axes.

(d) Repeat the previous question where a direction is assigned to each of the axes. Reversal of the direction of an axis is indicated by a minus sign; e.g., a rotation that reverses the z-axis and interchanges the x-axis and y-axis is written in cycle notation as

$$(x,y)(-x,-y)(z,-z).$$

11.3.3. The regular octahedron consists of eight equilateral triangles joined together so that the result looks like two pyramids joined base to base. A regular octahedron can be obtained by placing a vertex in each face of a cube and joining two vertices if they lie in faces which are separated by an edge.

(a) There is a duality between a regular octahedron and a cube in which faces correspond to vertices, edges to edges and vertices to faces. Obtain this correspondence.

(b) Write down the cycle index for the group of symmetries of the regular octahedron (reflections allowed or not) acting on the vertices of the regular octahedron.
Hint. This requires no calculation on your part.

(c) Do the same for the rotations of the octahedron acting on the edges.

11.3.4. Write down the cycle index for the group of rotations of the regular tetrahedron acting simultaneously on the vertices, edges and faces. Instead of the usual x_i, use v_i, e_i and f_i, indicating whether it is an orbit of vertices, faces or edges. For example, the identity rotation gives the term $v_1^4 e_1^6 f_1^4$. Explain how to use this result to obtain the cycle index for the group acting on just the edges.

11.3.5. Write down the cycle index polynomial for all permutations of $\underline{4}$ and use this to write down the ordinary generating function $D_4(y)$ for simple unlabeled 4-vertex digraphs by number of edges.

11.3.6. Repeat the previous exercise with "4" replaced by "5." You may find Exercises 2.2.5 and 2.3.3 (p. 57) helpful.

*11.3.7. State and prove a theorem like Theorem 11.9 for unlabeled n-vertex simple (loopless) graphs.
Hint. You will need to distinguish two cases depending on whether or not u and v are in the same cycle of g acting on $\underline{n}$.

11.4 Asymptotic Estimates

The area of asymptotics deals with obtaining estimates for functions for large values of the variables and, sometimes, for values near zero. Since the domain of the functions we're concerned with is the positive integers, these functions can be thought of as sequences $a_1, a_2, \ldots$. Since this section uses the terminology introduced in Appendix B, you may want to review it at this time.

A solid mathematical treatment of asymptotics requires more background than we are willing to assume and developing the background would take too much time. Therefore, the material in this section is not rigorous. Instead, we present several principles which indicate what the result will almost certainly be in common combinatorial situations. The intent of this section is to give you a feeling for the subject, some direction for future study and some useful rules of thumb.

Before launching into specific tools and examples, we'd like to set the stage a bit since you are probably unfamiliar with asymptotic estimates. The lack of specific examples may make some of this introductory material a bit vague, so you may want to reread it after completing the various subsections.

Suppose we are interested in a sequence of numbers. We have four methods of providing asymptotic information about the numbers. Here they are, with examples:

- A combinatorial description: say B_n is the number of partitions of an n-set;
- A recursion: $F_0 = 1$, $F_1 = 2$ and $F_n = F_{n-1} + F_{n-2}$ for $n \geq 2$;
- A formula: the number of involutions of an n-set is

$$\sum_{j=0}^{n} \frac{n!}{j!\, 2^j (n-2j)!};$$

the number of unlabeled full binary RP-trees with n leaves is $\frac{1}{n}\binom{2n-2}{n-1}$;

- A generating function: the ordinary generating function for the number of comparisons needed to Quicksort an n long list is
$$\frac{-2\ln(1-x)-2x}{(1-x)^2};$$
the ordinary generating function for the number of unlabeled rooted full binary trees by number of leaves satisfies
$$T(x) \;=\; \frac{T(x)^2+T(x^2)}{2}+x.$$

Given such information, can we obtain some information about the size of the terms in the sequence? The answer will, of course, depend on the information we are given. Here is a quick run down on the answers.

- A combinatorial description: It is usually difficult, if not impossible, to obtain information directly from such a description.
- A recursion: It is often possible to obtain some information. We will briefly discuss a simple case.
- A formula: The formula by itself may be explicit enough. If it is not, using Stirling's formula may suffice. If a summation is present, it can probably be estimated if all its terms are nonnegative, but it may be difficult or impossible to estimate a sum whose terms alternate in sign. Unfortunately, the estimation procedures usually involve a fair bit of messy calculation and estimation. We will discuss two common types of sums.
- A generating function: If the generating function converges for some values of x other than $x = 0$, it is quite likely that estimates for the coefficients can be obtained by using tools from analysis. Tools have been developed that can be applied fairly easily to some common situations, but rigorous application requires a background in complex analysis. The main emphasis of this section is the discussion of some simple tools for generating functions.

You may have noticed that we have discussed only singly indexed sequences in our examples. There are fewer tools available for multiply indexed sequences and they are generally harder to describe and to use. Therefore, we limit our attention to singly indexed sequences.

There is no single right answer to the problem of this section—finding simple approximations to some a_n for large n—we must first ask how simple and how accurate.

We will not try to specify what constitutes a simple expression; however, you should have some feel for it. For example, an expression of the form an^bn^c, where a, b and c are constants, is simple. The expression $\sqrt{2\pi n}(n/e)^n$ is simpler than the expression $n!$ even though the latter is more easily written down. Why? If we limit ourselves to the basic operations of addition, subtraction multiplication, division and exponentiation, then the former expression requires only six operations while $n!$ requires $n-1$ multiplications. We have hidden the work of multiplication by the use of a function, namely the factorial function. We can estimate simplicity by counting the number of basic operations.

There are wide variations in the degree of accuracy that we might ask for. Generally speaking, we would like an approximating expression whose relative error goes to zero. In other words, given a_n, we would like to find a simple expression $f(n)$ such that $a_n/f(n) \to 1$ as $n \to \infty$. In this case we say that a_n is *asymptotic to* $f(n)$ and write $a_n \sim f(n)$. Sometimes greater accuracy is desired—a problem we will not deal with. Sometimes we may have to settle for less accuracy—a situation we will be faced with.

The discussion of accuracy in the previous paragraph is a bit deceptive. What does $a_n \sim f(n)$ tell us about specific values? Nothing! It says that *eventually* $a_n/f(n)$ gets as close to 1 as we may desire, but eventually can be a *very* long time. But, in most cases of interest, we are lucky enough that the ratio $a_n/f(n)$ approaches 1 fairly quickly. Can we be more precise?

It is possible in most cases to compute an upper bound on how slowly $a_n/f(n)$ approaches 1; that is upper bounds on $|a_n/f(n) - 1|$ as a function of n. Obtaining such bounds often involves a considerable amount of work and is beyond the scope of this text. Even if one has a bound it may be unduly pessimistic—the ratio may approach one much faster than the bound says. A more pragmatic approach is to compute $a_n/f(n)$ for small values of n and hope that the trend continues for larger values. Although far from rigorous, this pragmatic approach almost always works well in practice. We'll carry out such calculations for the problems studied in this section.

The following subsections are independent of each other. If you are only going to read one of them, we recommend the one on generating functions.

Recursions

We have been able to solve the simplest sorts of recursions in earlier sections. Now our interest is different—we want asymptotic information from the recursions. We will consider two types of linear recursions that arise frequently in the analysis of algorithms. A *linear recursion* for a_n is an expression of the form

$$a_n = c_1(n)a_{n-1} + c_2(n)a_{n-2} + \ldots + c_n(n)a_0 + f(n)$$

for $n \geq N$. If $f(n) = 0$ for $n \geq N$, the recursion is called *homogeneous.*

We first discuss homogeneous recursions whose coefficients are "almost constant." In other words, except for initial conditions,

$$a_n = c_1(n)a_{n-1} + c_2(n)a_{n-2} + \ldots + c_k(n)a_{n-k}, \tag{11.33}$$

where the functions $c_i(n)$ are nearly constant. If $c_i(n)$ is nearly equal to C_i, then the solution to

$$A_n = C_1A_{n-1} + C_2A_{n-2} + \ldots + C_kA_{n-k}, \tag{11.34}$$

with initial conditions, should be reasonably close to the sequence a_n. We will not bother to discuss what reasonably close means here. It is possible to say something about it, but the subject is not simple.

What can we say about the solution to (11.34)? Without the initial conditions, we cannot say very much with certainty; however, the following is usually true.

Principle 11.1 Constant coefficient recursions *Let r be the largest root of the equation*

$$r^k = C_1r^{k-1} + C_2r^{k-2} + \ldots + C_kr^0. \tag{11.35}$$

If this root occurs with multiplicity m, then there is usually a constant A such that the solution to (11.34) that satisfies our (unspecified) initial conditions is asymptotic to $An^{m-1}r^n$.

This result is not too difficult to prove. Given the initial conditions, one can imagine using (11.34) to obtain a rational function for $\sum_n A_nx^n$. The denominator of the rational function will be $p(x) = 1 - (C_1x + \ldots + C_kx^k)$. Now imagine expanding the result in partial fractions. The reason for the lack of a guarantee in the principle is that a factor may cancel from the numerator and denominator of $\sum_n A_nx^n$, giving a lower degree polynomial than $p(x)$.

The principle is perhaps not so interesting because it gives much less accurate results than those obtained by using generating functions and partial fractions. Its only attractions are that it requires less work and gives us an idea of what to expect for (11.33).

Principle 11.2 Linear recursions *Suppose that $c_i(n) \to C_i$ as $n \to \infty$ and that at least one C_i is nonzero. Let r be the largest root of (11.35). Then r^n is probably a fairly reasonable crude approximation to the solution a_n of (11.33) that satisfies our (unspecified) initial conditions. Usually*

$$\lim_{n\to\infty} (a_n)^{1/n} = r.$$

Example 11.22 Involutions Let a_n be the number of permutations of $\underline{n}$ which are involutions, that is, no cycle lengths exceed two. Either n is in a cycle with itself OR it forms a cycle with one of the remaining $n-1$ elements of $\underline{n}$. Thus

$$a_n = a_{n-1} + (n-1)a_{n-2},$$

with some appropriate initial conditions.

The coefficients of this recursion are not almost constant, but we can use a trick, which works whenever we have coefficients which are polynomials in n. Let $b_n = a_n/(n!)^d$, where d is to be determined. Dividing our recursion by $(n!)^d$, and doing a little simple algebra, we have

$$b_n = \frac{1}{n^d} b_{n-1} + \frac{n-1}{(n^2-n)^d} b_{n-2}.$$

If $d < 1/2$, the last coefficient is unbounded, while if $d > 1/2$, both coefficients on the right side approach 0. On the other hand, with $d = 1/2$, the first coefficient approaches 0 and the second approaches 1. Thus we are led to consider the recursion $b_n = b_{n-2}$ and hence the roots of the polynomial $r^2 = 1$. Since the largest is $r = 1$, we expect that b_n should approach 1. Thus $(n!)^{1/2}$ is a rough approximation to a_n. We can eliminate the factorial by using Stirling's formula (Theorem 1.5 (p. 12)). Since the approximation in Principle 11.2 is so crude we may as well ignore factors like $\sqrt{2\pi n}$ and simply say that a_n probably grows roughly like $(n/e)^{n/2}$. $\square$

We now present a type of recursion that often arises in divide and conquer problems such as Mergesort. Some authors have called various forms of the theorem associated with this principle a master theorem for recursions. The reason we have put "function" in quotes is explained after the principle.

Principle 11.3 Master Principle for Recursions *We want to study the "function" $T(n)$. Suppose that for some "functions" $f(n), s_1(n), \ldots, s_w(n)$, some N, and some $0 < c < 1$ we have*

(i) $T(n) > 0$ *for* $n \ge N$,

(ii) $f(n) \ge 0$ *for* $n \ge N$,

(iii) $s_i(n) - cn \in O(1)$ *for* $1 \le i \le w$,

(iv) $T(n) = f(n) + T(s_1(n)) + T(s_2(n)) + \cdots + T(s_w(n))$.

Let $b = \log w / \log(1/c)$. Then

(a) *if $f(n)/n^b \to \infty$ as $n \to \infty$, then we usually have $T(n) \in \Theta(n^b)$;*

(b) *if $f(n)/n^b \to 0$ as $n \to \infty$, then we usually have $T(n) \in \Theta(f(n))$;*

(c) *if $n^b \in \Theta(f(n))$, then we usually have $T(n) \in \Theta(n^b \log n)$.*

The principle says that $T(n)$ grows like the faster growing of $f(n)$ and n^b unless they grow at the same rate, in which case $T(n)$ grows faster by a factor of $\log n$.

Why is "function" in quotes? Consider merge sorting. We would like $T(n)$ to be the number of comparisons needed to merge sort an n-long list. This is not a well-defined function! The number depends on the order of the items in the original list. (See Example 7.13 (p. 211).) Thus, we want to let $T(n)$ stand for the set of all possible values for the number of comparisons that are required. Hence $T(n)$ is really a collection of functions. Similarly $f(n)$ and, more rarely, the $s_i(n)$ may be collections of functions. Thus, a statement like $T(n) \in \Theta(f(n))$ should be interpreted as meaning that the statement is true for all possible values of $T(n)$ and $f(n)$.

Example 11.23 Recursive multiplication of polynomials Suppose we want to multiply two polynomials of degree at most n, say

$$P(x) \;=\; p_0 + p_1x + \cdots + p_nx^n \quad\text{and}\quad Q(x) \;=\; q_0 + q_1x + \cdots + q_nx^n.$$

A common method for doing this is to use the distributive law to generate $(n+1)^2$ products p_0q_0, p_0q_1x, $p_0q_2x^2$, …, $p_nq_nx^{2n}$ and then collect the terms that have the same powers of x. This involves $(n+1)^2$ multiplications of coefficients and, it can be shown, n^2 additions of coefficients. Thus, the amount of work is $\Theta(n^2)$.

Is this the best we can do? Of course, we can do better if $P(x)$ or $Q(x)$ have some coefficients that are zero. Since we're concerned with finding a general algorithm, we'll ignore this possibility. There is a recursive algorithm which is faster. It uses the following identity, which you should check

Identity: If $P_L(x)$, $P_H(x)$, $Q_L(x)$ and $Q_H(x)$ are polynomials, then

$$\begin{aligned}\bigl(P_L(x) + P_H(x)x^m\bigr)\bigl(Q_L(x) + Q_H(x)x^m\bigr)&\\ &= A(x) + \bigl(C(x) - A(x) - B(x)\bigr)x^m + B(x)x^{2m}\end{aligned}$$

where

$$A(x) \;=\; P_L(x)Q_L(x), \quad B(x) \;=\; P_H(x)Q_H(x),$$

and

$$C(x) \;=\; \bigl(P_L(x) + P_H(x)\bigr)\bigl(Q_L(x) + Q_H(x)\bigr).$$

We can think of this identity as telling us how to multiply two polynomials $P(x)$ and $Q(x)$ by splitting them into lower degree terms (the polynomials $P_L(x)$ and $Q_L(x)$) and higher degree terms (the polynomials $P_H(x)x^m$ and $Q_H(x)$):

$$P(x) \;=\; P_L(x) + P_H(x)x^m \quad\text{and}\quad Q(x) \;=\; Q_L(x) + Q_H(x)x^m.$$

The identity requires three polynomial multiplications to compute $A(x)$, $B(x)$ and $C(x)$. Since the degrees involved are only about half the degrees of the original polynomials, we've gone from about n^2 multiplications to about $3(n/2)^2 = 3n^2/4$, an improvement by a constant factor as $n \to \infty$. When this happens, applying an algorithm recursively usually gives an improvement in the exponent. In this case, we expect $\Theta(n^d)$ for some $d < 2$ instead of n^2.

Here's a recursive algorithm for multiplying two polynomials $P(x) = p_0 + p_1x \cdots + p_nx^n$ and $Q(x) = q_0 + q_1x + \cdots + q_nx^n$ of degree at most n.

MULT($P(x), Q(x), n$)
  If (n=0) Return p_0q_0
  Else
    Let $m = n/2$ rounded up.
    $P_L(x) \;=\; p_0 + p_1x + \cdots p_{m-1}x^{m-1}$
    $P_H(x) \;=\; p_m + p_{m+1}x + \cdots p_nx^{n-m}$
    $Q_L(x) \;=\; q_0 + p_1x + \cdots q_{m-1}x^{m-1}$
    $Q_H(x) \;=\; q_m + q_{m+1}x + \cdots q_nx^{n-m}$

```
        A(x) = MULT(P_L(x), Q_L(x), m - 1)
        B(x) = MULT(P_H(x), Q_H(x), n - m)
        C(x) = MULT(P_L(x) + P_H(x), Q_L(x) + Q_H(x), n - m)
        D(x) = A(x) + (C(x) - A(x) - B(x))x^m + B(x)x^{2m}
        Return D(x)
    End if
End
```

We store a polynomial as array of coefficients. The amount of work done in calculation is the number of times we multiply or add two coefficients.

Let's count multiplications. Since a polynomial of degree n has $n+1$ coefficients (constant term to coefficient of x^n), we'll denote by $T(n+1)$ the number of multiplications for a polynomial of degree n. To multiply two constants, we have $T(1) = 1$. You should be able to show that the recursive part of the algorithm gives us $T(n) = T(m)+T(n-m)+T(n-m)$, where $m = \lfloor n/2 \rfloor$, the largest integer not exceeding $n/2$. The Master Principle applies with $w = 3$, $f(n) = 0$, $s_1(n) = \lfloor n/2 \rfloor$, $s_2(n) = s_3(n) = n - \lfloor n/2 \rfloor = \lceil n/2 \rceil$[1] and $c = 1/2$. Thus $b = \log 3/\log 2$ and $T(n) \in \Theta(n^{\log 3/\log 2})$. Since $\log 3/\log 2$ is about 1.6 which is less than 2, this is less work than the n^2 multiplications in the common method when n is large.

What about additions and subtractions? The polynomials $A(x)$, $B(x)$ and $C(x)$ all have degree about n. Thus, computing $C(x) - A(x) - B(x)$ requires about $2n$ subtractions. The polynomials $A(x)$ and $(C(x) - A(x) - B(x))x^m$ overlap in about $n/2$ terms and so adding them together requires about $n/2$ additions. Adding in $B(x)x^{2m}$ requires about $n/2$ more additions. Thus, there are about $3n$ additions and subtractions involved in computing $D(x)$ from A, B and C. If $U(n)$ is the number of additions and subtractions in the recursive algorithm for polynomials of degree $n-1$,

$$U(n) = f(n) + U(m) = U(n-m) + U(n-m) \qquad \text{where} \qquad f(n) \in \Theta(3n).$$

By the Master Principle with $b = \log 3/\log 2$, $U(n) \in \Theta(n^b)$. Thus the algorithm requires $\Theta(n^{\log 3/\log 2})$ multiplications, additions and subtractions. □

Sums of Positive Terms

Suppose that

$$a_n = \sum_{k=0}^{L_n} t_{n,k},$$

where $t_{n,k} > 0$ and $L_n \to \infty$. Imagine n fixed and think of $t_{n,k}$ as a sequence in k; i.e., $t_{n,0}$,$t_{n,1}$, ... Let $r_k(n) = t_{n,k+1}/t_{n,k}$, the ratio of consecutive terms. Usually we simply write r_k for $r_k(n)$. In practice we usually have one of four situations

(a) **Decreasing terms** ($r_k \le 1$ for all k). We will study this.

(b) **Increasing terms** ($r_k \ge 1$ for all k) Convert to (a) by writing the sequence backwards:

$$a_n = \sum_{k=0}^{L_n} t_{n,k} = \sum_{i=0}^{L_n} t_{n,L_n-i} = \sum_{k=0}^{L_n} s_{n,k},$$

(c) **Increasing, then decreasing** ($r_k \le 1$ for $k < K_n$ and $r_k \ge 1$ for $k \ge K_n$): split the sum at K_n. This gives one sum like (a) and one like (b):

$$a_n = \sum_{k=0}^{L_n} t_{n,k} = \sum_{k=0}^{K_n-1} t_{n,k} + \sum_{k=0}^{M_n} u_{n,k},$$

[1] The ceiling function $\lceil x \rceil$ is the least integer not exceeding x.

where $M_n = L_n - K_n$ and $u_{n,k} = t_{n,k+K_n}$.

(d) **Decreasing, then increasing** ($r_k \geq 1$ for $k < K_n$ and $r_k \leq 1$ for $k \geq K_n$: Split into two as done for (c).

Suppose we are dealing with (a), decreasing terms, and that $\lim_{n\to\infty} r_k(n) = r$ exists for each k and does not depend on k. This may sound unusual, but it is quite common. If $r = 1$, we will call the terms *slowly decreasing.* If $|r| < 1$, we will call the terms *rapidly decreasing.* The two sums obtained from Case (c) are almost always slowly decreasing and asymptotically the same.

Principle 11.4 Sums of rapidly decreasing terms *If there is an r with $0 < r < 1$ such that $\lim_{n\to\infty} t_{n,k+1}/t_{n,k} = r$ for each value of k, then we usually have a geometric series approximation:*

$$\sum_{k=0}^{L_n} t_{n,k} \sim \sum_{k\geq 0} t_{n,0} r_n^k \sim \frac{t_{n,0}}{1-r}. \qquad 11.36$$

(Note that r does not depend on k.)

Aside: Actually, there is a more general principle: If $r_k(n) \to \rho_k$ as $n \to \infty$ and there is an $R < 1$ such that $|\rho_k| < R$ for all k, then the sum is asymptotic to $t_{n,0} \sum_{i=0}^{\infty} \rho_1 \cdots \rho_i$. Principle 11.4 is the special case $\rho_i = r$ for all i.

Example 11.24 Small subsets How many subsets of an n-set have at most cn elements where $c < 1/2$? You should have no trouble seeing that the answer is

$$\sum_{k=0}^{\lfloor cn \rfloor} \binom{n}{k},$$

where $\lfloor cn \rfloor$ is the largest integer that does not exceed cn. Let's approximate the sum. Since the terms are increasing, we reverse the order:

$$\sum_{k=0}^{\lfloor cn \rfloor} \binom{n}{\lfloor cn \rfloor - k}.$$

We have

$$\frac{t_{n,k+1}}{t_{n,k}} = \frac{\binom{n}{\lfloor cn \rfloor - k - 1}}{\binom{n}{\lfloor cn \rfloor - k}} = \frac{\lfloor cn \rfloor - k}{n - \lfloor cn \rfloor + k + 1}.$$

When k is small compared to n, this ratio is close to $c/(1-c)$ and so, by Principle 11.4, we expect

$$\sum_{k=0}^{\lfloor cn \rfloor} \binom{n}{k} \sim \frac{\binom{n}{\lfloor cn \rfloor}}{1 - (c/(1-c))} = \frac{1-c}{1-2c} \binom{n}{\lfloor cn \rfloor}. \qquad 11.37$$

A table comparing exact and approximate values is given in Figure 11.2. □

We now look at sums with slowly decreasing terms. Such sums can usually be done by interpreting the sum as an approximation to a Riemann sum associated with an integral. We'll merely state the result for the most common case.

c	n	10	20	50	100	200
0.1	A	11.25	213.75	2.384×10^6	1.947×10^{13}	1.815×10^{27}
	E	11	211	2.370×10^6	1.942×10^{13}	1.813×10^{27}
	A/E	1.023	1.01	1.006	1.003	1.002
0.2	A	60	6460	1.370×10^{10}	7.146×10^{20}	2.734×10^{42}
	E	56	6196	1.343×10^{12}	7.073×10^{20}	2.719×10^{42}
	A/E	1.07	1.04	1.02	1.01	1.005
0.4	A	630	377910	1.414×10^{14}	4.124×10^{28}	4.942×10^{57}
	E	386	263950	1.141×10^{14}	3.606×10^{28}	4.568×10^{57}
	A/E	1.6	1.4	1.2	1.14	1.08

Figure 11.2 The exact values (E) and approximate values (A) of the sum in (11.37) for $c = 0.1$, 0.2, 0.4 and $n = 10$, 20, 40, 100, 200. The ratios A/E are also given.

Principle 11.5 Sums of slowly decreasing terms *Let $r_k(n) = t_{n,k+1}/t_{n,k}$ and suppose $\lim_{n\to\infty} r_k(n) = 1$ for all k. Suppose there is a function $f(n) > 0$ with $\lim_{n\to\infty} f(n) = 0$ such that, for all "fairly large" k, $(1 - r_k(n))/k \sim f(n)$ as $n \to \infty$. Then we usually have*

$$\sum_{k\ge 0} t_{n,k} \sim \sqrt{\frac{\pi}{2f(n)}}\, t_{n,0}. \qquad 11.38$$

If the terms in the sum first increase and then decrease and the preceding applies to one half of the sum, then the answer in (11.38) should be doubled.

"Fairly large" means that k is small compared to n but that k is large compared with constants. For example, $(k-1)/kn$ can be replaced by $1/n$.

Example 11.25 Subsets of equal size Suppose j is constant. How many ways can we choose j distinct subsets of $\underline{n}$ all of the same size? Since there are $\binom{n}{k}$ subsets of size k, you should have no trouble seeing that the answer is

$$\sum_{k=0}^{n} \binom{\binom{n}{k}}{j}.$$

Since the binomial coefficients $\binom{n}{k}$ increase to a maximum at $k = \lfloor n/2 \rfloor$ and then decrease, the same is true for the terms in the sum. Thus we are in Case (c). We can take $K_n = \lfloor n/2 \rfloor$ and we expect (11.38) to apply.

For simplicity, let's treat n as if it is even. The second half of the sum is

$$\sum_{k=0}^{n/2} \binom{\binom{n}{k+n/2}}{j}$$

and so

$$\begin{aligned} r_k &= \binom{\binom{n}{k+1+n/2}}{j} \Big/ \binom{\binom{n}{k+n/2}}{j} \approx \left(\frac{\binom{n}{k+1+n/2}}{\binom{n}{k+n/2}}\right)^j \\ &= \left(\frac{n-k-n/2}{k+1+n/2}\right)^j \approx \left(\frac{1-2k/n}{1+2k/n}\right)^j \\ &\approx e^{-4jk/n} \approx 1 - 4jk/n, \end{aligned}$$

where the last approximations come from using $1+x \approx e^x$ as $x \to 0$, first with $x = \pm 2k/n$ and then with $x = 4jk/n$. Thus $(1-r_k)/k \approx 4j/n$. So we take $f(n) = 4j/n$ in Principle 11.5. Remembering that we need to double (11.38), we expect that

$$\sum_{k=0}^{n} \binom{\binom{n}{k}}{j} \sim 2\sqrt{\frac{\pi n}{8j}} \binom{\binom{n}{\lfloor n/2 \rfloor}}{j} \sim \sqrt{\frac{\pi n}{2j}} \frac{\binom{n}{\lfloor n/2 \rfloor}^j}{j!}.$$

This is correct. $\square$

Example 11.26 Involutions In Theorem 2.2 (p. 48), we showed that the number of involutions of $\underline{n}$ is

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

It is not obvious where the maximum term is; however, we can find it by solving the equation $t_{n,k+1}/t_{n,k} = 1$ for k. We have

$$\frac{t_{n,k+1}}{t_{n,k}} = \frac{(n-2k)(n-2k-1)}{2(k+1)} = 1. \tag{11.39}$$

Clearing fractions in the right hand equation leads to a quadratic equation for k whose solution is close to $m = (n - \sqrt{n})/2$. For simplicity, we assume that m is an integer. Since the maximum is not at the end, we expect to apply Principle 11.5. We split the sum into two pieces at m, one of which is

$$\sum_{k=0}^{\sqrt{n}} \frac{n!}{(n-2m-2k)!2^{m+k}(m+k)!}.$$

Adjusting (11.39) for this new index, we have

$$r_k(n) = \frac{t_{n,k+1}}{t_{n,k}} \approx \frac{(\sqrt{n}-2k)^2}{n-\sqrt{n}+2k}.$$

After some approximations and other calculations, we find that $r_k \approx 1 - (4k-1)/\sqrt{n}$. Thus $f(n) \sim 4/\sqrt{n}$ and so, doubling (11.38), we expect

$$I_n \sim \frac{2\sqrt{\pi/8}\, n^{1/4}\, n!}{(\sqrt{n})!\, 2^{(n-\sqrt{n})/2} \left((n-\sqrt{n})/2\right)!}.$$

If you wish, you can approximate this by using Stirling's formula. After some calculation, we obtain

$$I_n \sim \frac{n^{n/2}}{\sqrt{2}\, e^{n/2-\sqrt{n}+1/4}}. \tag{11.40}$$

This is correct. Here is a comparison of values.

n	5	20	50	100	200
(11.40)	23.6	2.241×10^{10}	2.684×10^{34}	2.340×10^{82}	3.600×10^{192}
I_n	26	2.376×10^{10}	2.789×10^{34}	2.405×10^{82}	3.672×10^{192}
ratio	0.91	0.94	0.96	0.97	0.98

The convergence is not very fast. It can be improved by making a more refined asymptotic estimate, which we will not do. $\square$

Example 11.27 Set partitions We proved in Theorem 11.3 (p. 320) that the number of partitions of an n-set equals

$$B_n = \frac{1}{e}\sum_{k=0}^{\infty}\frac{k^n}{k!}. \tag{11.41}$$

We'll use Principle 11.5 (p. 346). Taking ratios to find the maximum term:

$$\frac{t_{n,k+1}}{t_{n,k}} = \frac{(1+1/k)^n}{k+1} \approx \frac{e^{n/k}}{k}.$$

The approximation comes from $1+1/k = \exp(\ln(1+1/k)) \approx e^{1/k}$. We want $e^{n/k}/k = 1$. Taking logarithms, multiplying by k, and rearranging gives $k\ln k = n$. Let s be the solution to $s\ln s = n$. We split the sum at $\lfloor s \rfloor$. It is convenient (and does not affect the answer) to treat s as if it is an integer.

Unfortunately, the next calculations are quite involved. You may want to skip to the end of the paragraph. We now have a sum that starts at s and so, adjusting for shift of index,

$$r_k(n) = \frac{(1+1/(s+k))^n}{s+k+1} \approx \frac{e^{n/(s+k)}}{s+k} = \frac{e^{s\ln s/(s+k)}}{s+k}.$$

Since $\frac{1}{1+x} \approx 1-x$ for small x and since s is large, we have $\frac{s}{s+k} = \frac{1}{1+k/s} \approx 1-k/s$. Using this in our estimate for $r_k(n)$:

$$r_k(n) \approx \frac{e^{(1-k/s)\ln s}}{s+k} = \frac{se^{-(k\ln s)/s}}{s+k} \approx \left(1-\frac{k}{s}\right)\left(1-\frac{k\ln s}{s}\right),$$

where we have used $\frac{1}{1+x} \approx 1-x$ again and also $e^{-x} \approx 1-x$ for small x. When x and y are small, $(1-x)(1-y) \approx 1-(x+y)$. In this case, $x = k/s$ is much smaller than $y = k(\ln s)/s$ so that $x+y \approx y$ and so we finally have

$$r_k(n) \approx 1 - \frac{k\ln s}{s}$$

and so $f(n) = (\ln s)/s$.

Remembering to double and to include the factor of $1/e$ from (11.41), we have (using Stirling's formula on $s!$)

$$B_n \sim \frac{\sqrt{2\pi s/\ln s}\, s^n}{e\; s!} \sim \frac{s^{n-s}e^{s-1}}{\sqrt{\ln s}}, \quad \text{where } s\ln s = n \text{ defines } s. \tag{11.42}$$

Here's how (11.42) compares with the exact values.

n	5	20	50	100	200
s	3.76868	9.0703	17.4771	29.5366	50.8939
(11.42)	70.88	6.305×10^{13}	2.170×10^{47}	5.433×10^{115}	7.010×10^{275}
B_n	52	5.172×10^{13}	1.857×10^{47}	4.759×10^{115}	6.247×10^{275}
ratio	1.36	1.22	1.17	1.14	1.22
better	1.03	1.01	1.005	1.003	1.002

The approximation is quite poor. Had we used the factor of $\sqrt{1+\ln s}$ in the denominator of (11.42), the relative error would have been much better, as shown in the last line of the table. How did we obtain the formula with $\sqrt{1+\ln s}$? After obtaining the form with $\sqrt{\ln s}$ and noting how poor the estimate was, we decided to look for a correction by trial and error. Often, a good way to do this is by adjusting in a simple manner the part of the estimate that contains the smallest function of n—in this case, the function $\ln s$. We tried $C+\ln s$ and found that $C = 1$ gave quite accurate estimates. □

Generating Functions

In order to study asymptotic estimates from generating functions, it is necessary to know something about singularities of functions. Essentially, a singularity is a point where the function misbehaves in some fashion. The singularities that are encountered in combinatorial problems are nearly all due to either

- attempting to take the logarithm of zero,
- attempting to raise zero to a power which is not a positive integer,

or both. For example, $-\ln(1-x)$ has a singularity at $x = 1$. The power of zero requires a bit of explanation. It includes the obviously bad situation of attempting to divide by zero; however, it also includes things like attempting to take the square root of zero. For example, $\sqrt{1-4x}$ has a singularity at $x = 1/4$. To explain why a nonintegral power of zero is bad would take us too far afield. Suffice it to say that the fact that A has two square roots everywhere except at $A = 0$ is closely related to this problem.

The following is stated as a principle because we need to be more careful about the conditions in order to have a theorem. For combinatorial problems, you can expect the more technical conditions to be satisfied.

Principle 11.6 Nice singularities *Let a_n be a sequence whose terms are positive for all sufficiently large n. Suppose that $A(x) = \sum_n a_n x^n$ converges for some value of $x > 0$. Suppose that $A(x) = f(x)g(x) + h(x)$ where*

- *$f(x) = \big(-\ln(1-x/r)\big)^b (1-x/r)^c$, c is not a positive integer and we do not have $b = c = 0$;*
- *$A(x)$ does not have a singularity for $-r \le x < r$;*
- *$\lim_{x\to r} g(x)$ exists and is nonzero (call it L);*
- *$h(x)$ does not have a singularity at $x = r$.*

Then it is usually true that

$$a_n \sim \begin{cases} \dfrac{L(\ln n)^b (1/r)^n}{n^{c+1}\Gamma(-c)}, & \text{if } c \neq 0; \\[2ex] \dfrac{bL(\ln n)^{b-1}(1/r)^n}{n}, & \text{if } c = 0; \end{cases} \qquad 11.43$$

where Γ is the **Gamma function** *which we describe below.*

The value of r can be found by looking for the smallest (positive) singularity of $A(x)$. Often $g(r)$ is defined and then $g(r) = L$. Since, in many cases there is no logarithm term (and so $b = 0$), you may find it helpful to rewrite the principle for the special case $b = 0$.

The values of the Gamma function $\Gamma(x)$ can be looked up in tables. In practice, the only information you are likely to need about this function is

$$\Gamma(k) = (k-1)! \quad \text{when } k > 0 \text{ is an integer}, \quad \Gamma(x+1) = x\Gamma(x) \quad \text{and} \quad \Gamma(1/2) = \sqrt{\pi}.$$

For example, we compute $\Gamma(-1/2)$ by using $\Gamma(1/2) = (-1/2)\Gamma(-1/2)$:

$$\Gamma(-1/2) = -2\Gamma(1/2) = -2\sqrt{\pi}.$$

We begin with a couple of simple examples.

Example 11.28 Derangements Let D_n be the number of permutations of $\underline{n}$ that have no fixed points. We showed in (11.17) that

$$\sum_{n=0}^{\infty} \frac{D_n x^n}{n!} \;=\; \frac{e^{-x}}{1-x}.$$

We apply Principle 11.6 with this as $A(x)$. There is a singularity at $x = 1$ because we are then dividing by zero. Thus $r = 1$ and we have

$$A(x) \;=\; (1-x)^{-1}e^{-x} + 0, \qquad \text{so} \quad f(x) = (1-x)^{-1}, \quad g(x) = e^{-x}, \quad \text{and} \quad h(x) = 0.$$

Thus $b = 0$ and $c = -1$. Since $g(1) = 1/e$, we have $D_n/n! \sim 1/e$. In Example 4.5 (p. 99) we used an explicit formula for D_n to show that D_n is the closest integer to $n!/e$. We get no such error estimate with this approach. □

Example 11.29 Rational generating functions Suppose that $A(x) = \sum_n a_n x^n = p(x)/q(x)$ where $a_n \geq 0$ and $p(x)$ and $q(x)$ are polynomials such that

- r is the smallest zero of $q(x)$; that is, $q(r) = 0$ and $q(s) \neq 0$ if $0 < s < r$;
- the multiplicity of r as a zero is k; that is, $q(x) = s(x)(1-x/r)^k$ where $s(x)$ is a polynomial and $s(r) \neq 0$;
- $p(r) \neq 0$;

We can apply Principle 11.6 with $f(x) = (1-x/r)^{-k}$ and $g(x) = p(x)/s(x)$. Since r is a zero of $q(x)$ of multiplicity k, it follows that this gives an asymptotic formula for a_n:

$$a_n \;\sim\; \frac{p(r)n^{k-1}(1/r)^n}{s(r)\,(k-1)!}. \tag{11.44}$$

We leave it to you to apply this formula to various rational generating functions that have appeared in the text. □

Example 11.30 The average time for Quicksort Let q_n be the average number of comparisons that are needed to Quicksort n long lists. In Example 10.12 (p. 289) we obtained the ordinary generating function

$$Q(x) \;=\; \frac{-2\ln(1-x) - 2x}{(1-x)^2}.$$

We can take $A(x) = Q(x)$, $r = 1$, $f(x) = -\ln(1-x)\,(1-x)^{-2}$ and $g(x) = 2 + \big(2x/\ln(1-x)\big)$. Then $\lim_{x\to 1} g(x) = 2$. From (11.43),

$$q_n \;\sim\; 2\binom{n+1}{n}\ln n \sim 2n\ln n, \tag{11.45}$$

which we also obtained in (10.30) by manipulating an explicit expression for q_n. Here are some numerical results.

n	5	20	50	100	1000
(11.45)	16.09	119.8	391.2	921.0	13816.
q_n	7.4	71.11	258.9	647.9	10986.
ratio	2.2	1.7	1.5	1.4	1.3

The approximation converges very slowly. You might like to experiment with adjusting (11.45) to improve the estimate.

An alternative approach is to write $Q(x)$ as a difference of two functions and deal with each separately: $Q(x) = Q_1(x) - Q_2(x)$ where

$$Q_1(x) \;=\; \frac{-2\ln(1-x)}{(1-x)^2} \qquad \text{and} \qquad Q_2(x) \;=\; \frac{2x}{(1-x)^2}.$$

Now $f_1(x) = -\ln(1-x)(1-x)^{-2}$, $f_2(x) = (1-x)^{-2}$, $g_1(x) = 2$, $g_2(x) = 2x$ and $h_1(x) = h_2(x) = 0$. We obtain $q_{1,n} \sim 2n \ln n$ as before and $q_{2,n} \sim 2n$. Subtracting we again obtain $q_n \sim 2n \ln n$. $\square$

So far we have dealt with generating functions that required relatively little algebra to obtain the asymptotics. We now look at a more complicated situation.

Example 11.31 Binary operations In Example 11.3 (p. 312) we studied z_n, the number of ways a string of n zeroes can be parenthesized to give zero, and obtained the ordinary generating function

$$Z(x) \;=\; \frac{T(x) - 1 + \sqrt{\big(1-T(x)\big)^2 + 4x}}{2},$$

where

$$T(x) \;=\; \frac{1-\sqrt{1-4x}}{2}$$

is the ordinary generating function for all ways to parenthesize the string. (See (11.12) and (11.14).)

To gear up for studying $Z(x)$, we begin with the simple $T(x)$. Of course, $T(x)$ could easily be studied by using the explicit formula for its coefficients, $t_n = \frac{1}{n}\binom{2n-2}{n-1}$; however, the point is to understand how to handle the square root singularity. The square root has a singularity at $r = 1/4$ since it vanishes there. Thus we write

$$T(x) \;=\; f(x)g(x) + h(x), \qquad \text{where} \quad f(x) = (1-x/r)^{1/2}, \quad g(x) = -1/2 \quad \text{and} \quad \text{h(x)=1/2}.$$

From (11.43) we obtain

$$\begin{aligned} t_n \;&=\; a_n \sim (-1/2)\binom{n-3/2}{n}(1/4)^{-n} \\ &\sim\; (-1/2)\Big(n^{-3/2}/\Gamma(-1/2)\Big)4^n \;=\; \frac{4^{n-1}}{\sqrt{\pi}\, n^{3/2}}, \end{aligned} \qquad 11.46$$

since $(-1/2)\Gamma(-1/2) = \Gamma(1/2) = \sqrt{\pi}$.

We're ready for $Z(x)$. Since $r = 1/4$ is a singularity of $T(x)$, it is also a singularity of $Z(x)$. We can have other singularities when the complicates square root is zero; that is, $(1-T(x)^2) + 4x = 0$. We have

$$(1-T(x))^2 + 4x \;=\; \frac{1 + 2\sqrt{1-4x} + (1-4x)}{4} + 4x \;=\; \frac{1 + 6x + \sqrt{1-4x}}{2}.$$

For this to be zero we must have

$$1 + 6x \;=\; -\sqrt{1-4x}. \qquad 11.47$$

Squaring: $1 + 12x + 36x^2 = 1 - 4x$ and so $16x + 36x^2 = 0$. The two solutions are $x = 0$ and $x = -4/9$. Since squaring can introduce false solutions, so we'll have to check them in (11.47). The value $x = 0$ is not a solution to (11.47) and the value $x = -4/9$ is irrelevant since it does not lie in the interval $[-1/4, 1/4]$. Thus $r = 1/4$.

How can we find f, g and h? It seems likely that $f(x) = \sqrt{1-4x}$ since we have $T(x)$ present in the numerator of $Z(x)$. Then $Z(r) = f(r)g(r) + h(r) = h(r)$ since $f(1/4) = 0$. We could simply try to let $h(x)$ be the constant $Z(r)$, which is $(-1+\sqrt{5}\,)/4$. Thus we have

$$f(x) \;=\; \sqrt{1-4x}, \qquad h(x) \;=\; Z(1/4) \;=\; \frac{\sqrt{5}-1}{4}, \qquad g(x) \;=\; \frac{Z(x) - Z(1/4)}{\sqrt{1-4x}}.$$

We need to find L. To simplify expressions, let $s = \sqrt{1-4x}$. We have

$$L \;=\; \lim_{x\to 1/4}\left(\frac{-s + \sqrt{(1+s)^2 + 16x} - \sqrt{5}}{4s}\right). \qquad 11.48$$

n	5	10	20	30	40	50
(11.46)	12.92	4676.	1.734×10^9	9.897×10^{14}	6.740×10^{20}	5.057×10^{26}
t_n	14	4862	1.767×10^9	1.002×10^{15}	6.804×10^{20}	5.096×10^{26}
t ratio	0.92	0.96	0.98	0.987	0.991	0.992
(11.49)	3.571	1293.	4.792×10^8	2.735×10^{14}	1.863×10^{20}	1.398×10^{26}
z_n	5	1381	4.980×10^8	2.806×10^{14}	1.899×10^{20}	1.419×10^{26}
z ratio	0.7	0.94	0.96	0.97	0.98	0.985
z_n/t_n	0.36	0.284	0.282	0.280	0.279	0.279

Figure 11.3 Asymptotic and exact values for t_n and z_n in Example 11.31. The ratios of asymptotic to exact values are given and the ratio z_n/t_n, which we have shown should approach $(5-\sqrt{5})/10 = 0.276393$.

How can we evaluate the limit? Since numerator and denominator approach zero, we can apply l'Hôpital's Rule (a proof can be found in any rigorous calculus text):

Theorem 11.10 l'Hôpital's Rule *Suppose that* $\lim_{x\to a} f(x) = 0$, $\lim_{x\to a} g(x) = 0$, *and* $g(x) \neq 0$ *when* $0 < |x-a| < \delta$ *for some* $\delta > 0$. *Then*

$$\lim_{x\to a} \frac{f(x)}{g(x)} = \lim_{x\to a} \frac{f'(x)}{g'(x)},$$

provided the latter limit exists.

We leave it to you to apply l'Hôpital's Rule and some algebra to evaluate the limit. We'll use a trick—rewrite everything in terms of s and then use l'Hôpital's Rule:

$$\begin{aligned} L &= \lim_{s\to 0}\left(\frac{-s+\sqrt{(1+s)^2-4(s^2-1)}-\sqrt{5}}{4s}\right) && \text{by algebra} \\ &= \lim_{s\to 0}\left(\frac{-s+\sqrt{5+2s-3s^2}-\sqrt{5}}{4s}\right) && \text{by algebra} \\ &= \lim_{s\to 0}\left(\frac{-1+(1/2)(5+2s-3s^2)^{-1/2}(2-6s)}{4}\right) && \text{by l'Hôpital's Rule} \\ &= \frac{-1+5^{-1/2}}{4} = -\frac{5-\sqrt{5}}{20}. \end{aligned}$$

We finally have

$$z_n \sim -\frac{5-\sqrt{5}}{20}\binom{n-3/2}{n}(1/4)^{-n} \sim \frac{5-\sqrt{5}}{10}\frac{4^{n-1}}{\sqrt{\pi}\,n^{3/2}}. \qquad 11.49$$

Comparing the result with the asymptotic formula for t_n, the total number of ways to parenthesize the sequence $0 \wedge \ldots \wedge 0$, we see that the fraction of parenthesizations that lead to a value of zero approaches $(5-\sqrt{5})/10$ as $n \to \infty$. Various comparisons are shown in Figure 11.3. $\square$

Example 11.32 Three misfits Although Principle 11.6 applies to many situations of interest in combinatorics, there are also many cases where it fails to apply. Here are three examples.

- No singularities: The EGF for the number of involutions of $\underline{n}$ is $\exp(x + x^2/2)$. This function has no singularities, so our principle fails.
- Zero radius of convergence: The number of simple graphs with vertex set $\underline{n}$ is $g_n = 2^N$ where $N = \binom{n}{2}$. By the Exponential Formula (Theorem 11.5 (p. 321)), the EGF for the number of connected graphs is $\ln(G(x))$, where $G(x)$ is the EGF for g_n. Unfortunately, $G(x)$ only converges at $x = 0$ so our principle fails.
- Bad singularities: Let $p(n)$ be the number of partitions of the integer n. At the end of Example 10.15 (p. 295) we showed that the ordinary generating function for $p(n)$ is
$$P(x) \;=\; \prod_{i=1}^{\infty}(1 - x^i)^{-1}.$$
Clearly $r = 1$. Unfortunately, for every real number c, $P(x)/(1-x)^c \to \infty$ as $x \to 1$. Thus our principle does not apply; however, asymptotic information about $p(n)$ can be deduced from the formula for $P(x)$. The methods are beyond this text. (Actually, $P(x)$ behaves even worse than we indicated—it has a singularity for all x on the unit circle.) □

So far we have dealt with generating functions that are given by an explicit formula. This does not always happen. For example, the ordinary generating function for rooted unlabeled full binary trees by number of leaves is given implicitly by
$$T(x) \;=\; \frac{T(x)^2 + T(x^2)}{2} + x.$$
Although any specific t_n can be found, there is no known way to solve for the function $T(x)$. The following principle helps with many such situations. Unfortunately, the validity of its conclusion is a bit shakier than (11.43).

Principle 11.7 Implicit functions *Let a_n be a sequence whose terms are positive for all sufficiently large n. Let $A(x)$ be the ordinary generating function for the a_n's. Suppose that the function $F(x,y)$ is such that $F(x, A(x)) = 0$. If there are positive real numbers r and s such that $F(r,s) = 0$ and $F_y(r,s) = 0$ and if r is the smallest such r, then it is usually true that*
$$a_n \;\sim\; \sqrt{\frac{rF_x(r,s)}{2\pi F_{yy}(r,s)}}\, n^{-3/2} r^{-n}. \qquad 11.50$$

Example 11.33 RP-trees with degree restrictions Let D be a set of nonnegative integers that contains 0. In Exercise 10.4.11 (p. 301), we said an RP-tree was of outdegree D if the number of sons of each vertex lies in D. Let $T_D(x)$ be the ordinary generating function for unlabeled RP-trees of outdegree D by number of vertices. In Exercise 10.4.11, you were asked to show that
$$T_D(x) \;=\; x \sum_{d \in D} T_D(x)^d.$$

It can be shown that $t_D(n) > 0$ for all sufficiently large n if and only if
$$\gcd\{d - 1 : d \in D\} \;=\; 1.$$
We invite you to prove this by looking at the equation for $g(x) = T_D(x)/x$ and using the fact that all sufficiently large multiples of the gcd of a set S can be expressed as a nonnegative linear combination of the elements of S.

Let $F(x,y) = y - x\sum_{d\in D} y^d$. Then r and s in Principle 11.7 are found by solving the equations

$$\begin{aligned} s - r\sum_{d\in D} s^d &= 0, \\ 1 - r\sum_{d\in D} ds^{d-1} &= 0. \end{aligned}$$

With a bit of algebra, we can get an equation in s alone which can be solved numerically and then used in a simple equation to find r:

$$1 = \sum_{\substack{d\in D\\ d>0}} (d-1)s^d, \tag{11.51}$$

$$r = \left(\sum_{d\in D} s^{d-1}\right)^{-1}. \tag{11.52}$$

Finally

$$\frac{rF_x(r,s)}{F_{yy}(r,s)} = \frac{\sum_{d\in D} s^d}{\sum_{d\in D} d(d-1)s^{d-2}}. \tag{11.53}$$

Since the right side of (11.51) is a sum of positive terms, it is a simple matter to solve it for the unique positive solution to any desired accuracy. This result can then be used in (11.52) to find r accurately. Finally, the results can be used in (11.53) and (11.50) to estimate $t_D(n)$. $\square$

Example 11.34 Unlabeled rooted trees In (11.29) we showed that the generating function by vertices for unlabeled rooted trees in which each vertex has outdegree at most three satisfies

$$T(x) = 1 + x\frac{T(x)^3 + 3T(x)T(x^2) + 2T(x^3)}{6}. \tag{11.54}$$

Since $T(x^2)$ and $T(x^3)$ appear here, it does not seem possible to apply Principle 11.7. However, it can be applied—we simply set

$$F(x,y) = 1 - y + x\frac{y^3 + 3T(x^2)y + 2T(x^3)}{6}.$$

The reason why this is permitted is somewhat technical: The singularity r of $T(x)$ turns out to lie in $(0,1)$, which guarantees that $x^3 < x^2 < x$ when x is near r. As a result $T(x^2)$ and $T(x^3)$ are not near a singularity when x is near r.

Even if you did not fully follow the last part of the previous paragraph, you may have noticed the claim that the singularity of $T(x)$ lies in $(0,1)$. We should prove this, but we will not do so. Instead we will be satisfied with noting that our calculations produce a value of $r \in (0,1)$—a circular argument since we needed that fact before beginning.

The equations for r and s are

$$1 - s + r\frac{s^3 + 3T(r^2)s + 2T(r^3)}{6} = 0, \tag{11.55}$$

$$-1 + r\frac{s^2 + T(r^2)}{2} = 0. \tag{11.56}$$

We have a problem here: In order to compute r and s we need to be able to evaluate the function T accurately and we lack an explicit formula for $T(x)$. This can be gotten around as follows.

Suppose we want to know $T(x)$ at some value of $x = p \in (0,1)$. If we knew the values of $T(p^2)$ and $T(p^3)$, we could regard (11.54) as a cubic in the value of $T(p)$ and solve it. On the other hand, if we knew the values of $T((p^2)^2)$ and $T((p^2)^3)$, we could set $x = p^2$ in (11.54) and solve the resulting cubic for $T(p^2)$. On the surface, this does not seem to be getting us anywhere because we keep needing to know $T(p^k)$ for higher and higher powers k. Actually, that is not what we need—we need to know $T(p^k)$ *with some desired degree of accuracy.* When k is large, p^k is close to zero and so $T(p^k)$ is close to $T(0) = t_0 = 1$. (We remind you that when we derived (11.54) we chose to set $t_0 = 1$.) How large does k need to be for a given accuracy? We won't go into that.

There is another trick that will simplify our work further. Since $s = T(r)$, we can eliminate s from (11.56) and so $r\big(T(r)^2 + T(r^2)\big) = 1$ is the equation for r. We can now use (11.55) to check for errors in our calculations.

Once r and s have been found it is a fairly straightforward matter to apply (11.50). The only issue that may cause some difficulty is the evaluation of

$$F_x(r,s) \;=\; \frac{s^3 + 3T(r^2)s + 2T(r^3)}{6} + r^3T'(r^2)s + r^3T'(r^3)$$

because of the presence of T'. We can differentiate (11.54) with respect to x and solve the result for $T'(x)$ in terms of x, $T'(x^2)$, $T'(x^3)$ and values of T. Using this recursively we can evaluate T' to any desired degree of accuracy, much as we can T. After considerable calculation, we find that

$$t_n \;\sim\; (0.51788\ldots)n^{-3/2}(0.3551817\ldots)^{-n} \qquad 11.57$$

which can be proved. Since we have the results

n	5	10	20	30	40	50
(11.57)	8.2	512.5	5671108.	9.661×10^{10}	1.964×10^{15}	4.398×10^{19}
t_n	8	507	5622109	9.599×10^{10}	1.954×10^{15}	4.380×10^{19}
ratio	1.02	1.01	1.009	1.006	1.005	1.004

the estimate is a good approximation even for small n. $\square$

Exercises

In this set of exercises, "estimate" always means asymptotically; i.e., for large n. Since you will be using the various principles in the text, your results will only be what is probably true; however, the results obtained will, in fact, be correct. Some problems ask you to use alternate methods to obtain asymptotic estimates. We recommend that, after doing such a problem, you reflect on the amount of work each method requires and the accuracy of the result it produced. In many of the exercises, enumeration formulas are given. You need not derive them unless you are asked to do so.

11.4.1. A path enumeration problem leads to the recursion $a_n = 2a_{n-1} + a_{n-2}$ for $n \geq 2$ with initial conditions $a_0 = 1$ and $a_1 = 3$.

(a) Estimate a_n directly from the recursion using Principle 11.1 (p. 341).

(b) Determine the ordinary generating function and use (11.44) to estimate a_n.

(c) Use the ordinary generating function to obtain an explicit formula for a_n and use this formula to estimate a_n.

11.4.2. The recursions

$$\begin{aligned} U_n &= nU_{n-1} + 2U_{n-2} - (n-4)U_{n-3} - U_{n-4} \\ V_n &= (n-1)(V_{n-1} + V_{n-2}) + V_{n-3} \end{aligned}$$

arise in connection with the "menage" problems. Estimate U_n and V_n from the recursions.

11.4.3. In Example 7.13 (p. 211), an upper bound was found for the running time to merge sort a 2^k-long list. Show that the running time to merge sort an n-long list is $\Theta(n \log n)$.
Note: This is for general n, not just $n = 2^k$ and it is for *any* list, not just worst case.

11.4.4. Let a_n be the number of permutations f of $\underline{n}$ such that $f^k(x) = x$ for all $x \in \underline{n}$.

(a) Show that

$$a_n = \sum_{d|k} \binom{n-1}{d-1} (d-1)!\, a_{n-d},$$

where "$d|k$" beneath the summation sign—read "d divides k"—means that the sum is over all positive integers d such that k/d is an integer.

(b) Estimate a_n from the recursion.

11.4.5. A *functional digraph* is a simple digraph in which each vertex has outdegree 1. (See Exercise 11.2.17.) We say it is connected if the associated graph is connected.

(a) The number of labeled connected n-vertex functional digraphs is

$$\sum_{k=1}^{n} \frac{n!\, n^{n-k+1}}{(n-k)!}.$$

Obtain the estimate $\sqrt{\pi n/2}\, n^{n+1}$ for this summation.

(b) The average number of components in a labeled n-vertex functional digraph is

$$\sum_{k=1}^{n} \frac{n!}{k\, n^k (n-k)!}.$$

Obtain an estimate for this summation.

11.4.6. Let a_n be the number of partitions of $\underline{n}$ into ordered blocks.

(a) Show that $\sum_n a_n x^n/n! = (2 - e^x)^{-1}$

(b) Estimate a_n from the generating function.

*(c) By expanding $(1 - e^x/2)^{-1}$, show that

$$a_n = \sum_{k=1}^{\infty} \frac{k^n}{2^{k+1}}$$

and use this summation to estimate a_n.

11.4.7. Let S be a set of positive integers and let a_n be the number of permutations f of $\underline{n}$ such that none of the cycles of f have a length in S. Then

$$\sum_{n\geq 0} a_n x^n/n! = \frac{1}{1-x} \exp\left(-\sum_{k\in S} x^k/k\right).$$

When S is a finite set, estimate a_n.

11.4.8. Let a_n be the number of labeled simple n-vertex graphs all of whose components are cycles.

(a) Show that

$$\sum_{n\geq 0} a_n x^n/n! = \frac{\exp(-x/2 - x^2/4)}{\sqrt{1-x}}.$$

(b) Obtain an estimate for a_n.

11.4.9. The EGFs for permutations all of whose cycles are odd is $A_o(x) = \sqrt{\frac{1+x}{1-x}}$, and that for permutations all of whose cycles are even is $A_e(x) = (1 - x^2)^{-1/2}$.

(a) Why can't our principles for generating functions be used to estimate the coefficients of $A_o(x)$ and $A_e(x)$?

(b) If you apply the relevant principle, it will give the right answer anyway for $A_o(x)$ but not for $A_e(x)$. Apply it.

(c) Show that $a_{e,2n+1} = 0$ and $a_{e,2n} = \binom{2n}{n}(2n)!\, 4^{-n}$ and then use Stirling's formula.

(d) Use $A_o(x) = (1 + x)(1 - x^2)^{-1/2}$ to obtain formulas for $a_{o,n}$.

11.4.10. Let S be a set of positive integers and let S' be those positive integers not in S. Let $c_n(S)$ be the number of compositions of n all of whose parts lie in S, with $c_0(S) = 1$.

(a) Derive the formula

$$\sum_{n=0}^{\infty} c_n(S)x^n = \frac{1}{1 - \sum_{k \in S} x^k}.$$

(b) Explain how to derive asymptotics for $c_n(S)$ when S is a finite set.

(c) Explain how to derive asymptotics for $c_n(S)$ when S' is a finite set.

11.4.11. A "path" of length n is a sequence $0 = u_0, u_1, \ldots, u_n = 0$ of nonnegative integers such that $u_{k+1} - u_k \in \{-1, 0, 1\}$ for $k < n$. The ordinary generating function for such paths by length is

$$\frac{1}{\sqrt{1 - 2x - 3x^2}}.$$

Estimate the number of such paths by length. (See Exercise 10.3.2.)

11.4.12. In Exercise 10.4.7 (p. 300) we showed that the generating function for an unlabeled binary RP-tree by number of vertices is $(1 - x - \sqrt{1 - 2x - 3x^2})/2x$. Estimate the coefficients.

11.4.13. A certain kind of unlabeled RP-trees have the generating function

$$\frac{1 + x^2 - \sqrt{(1 + x^2)^2 - 4x}}{2}$$

when counted by vertices. Estimate the number of such trees with n vertices.

11.4.14. Show that the EGF for permutations with exactly k cycles is

$$\frac{1}{k!}\left\{\ln\left(\frac{1}{1 - x}\right)\right\}^k$$

and use this to estimate the number of such permutations when k is fixed and n is large.

11.4.15. Let h_n be the number of unlabeled n-vertex RP-trees in which no vertex has outdegree 1 and let $H(x)$ be the ordinary generating function. Show that

$$H(x)^2 - H(x) + \frac{x}{1 + x} = 0$$

and use this to estimate h_n.

11.4.16. Let h_n be the number of rooted labeled n-vertex trees for which no vertex has outdegree 2. It can be shown that the EGF $H(x)$ satisfies $(1 + x)H(x) = xe^{H(x)}$. Estimate h_n.

11.4.17. Let D be a set of positive integers. Let a_n be the number of functions f from $\underline{n}$ to the positive integers such that, (a) if f takes on the value k, then it takes on the value i for all $0 < i < k$, and (b) if f takes on the value k the number of times it does so lies in D. It can be shown that the EGF for the a_n's is

$$A(x) = \left(1 - \sum_{k \in D} x^k/k!\right)^{-1}$$

For whatever D's you can, show how to estimate a_n.

11.4.18. Let the ordinary generating function for the number of rooted unlabeled n-vertex trees in which every vertex has outdegree at most 2 be $T(x)$. For convenience, set $t_0 = 1$. It can be shown that $T(x) = 1 + x\big(T(x)^2 + T(x^2)\big)/2$. Estimate the numbers of such trees.

*11.4.19. The results here relate to Exercise 10.4.1 (p. 298). We suppose that $A(x)$ satisfies Principle 11.6 and that $a_0 = 0$. Our notation will be that of Principle 11.6 and we assume that $b = 0$ for simplicity.

(a) Suppose that $c < 0$. Let $B(x) = A(x)^k$. Show that we expect $b_n/a_n \sim (g(r)n^{-c})^{k-1}\Gamma(-c)/\Gamma(-ck)$.

(b) Suppose that $c > 0$ and $h(r) \neq 0$. Let $B(x) = A(x)^k$. Show that we expect $b_n/a_n \sim kh(r)^{k-1}$.

(c) Suppose that $c = 1/2$ and $A(r) < 1$. Let $B(x) = (1 - A(x))^{-1}$. Derive

$$B(x) \;=\; \frac{1 - h(x) + f(x)g(x)}{(1 - h(x))^2 - (1 - x/r)g(x)^2}$$

and use this to show that we expect $b_n/a_n \sim (1-h(r))^{-2}$. You may assume that the denominator $(1 - h(x))^2 - (1 - x/r)g(x)^2$ does not vanish on the interval $[-r, r]$.

(d) Suppose that $A(x) = 1$ has a solution $s \in (0, r)$. Show that it is unique. Let $B(x) = (1-A(x))^{-1}$. Show that we expect $b_n \sim 1/(A'(s)s^{n+1})$. Prove that the solution $A(s) = 1$ will surely exist if $c < 0$.

(e) Suppose $r < 1$. It can be shown that the radii of convergence of

$$\sum_{k\ge 2} A(x^k)/k \qquad \text{and} \qquad \sum_{k\ge 2}(-1)^{k-1}A(x^k)/k$$

both equal 1. Explain how we could use this fact to obtain asymptotics for sets and multisets in Exercise 10.4.1 (p. 298) using Principle 11.6, if we could handle $e^{A(x)}$ using the principle.

11.4.20. Recall that the generating function for unlabeled full binary RP-trees by number of leaves is

$$\frac{1 - \sqrt{1 - 4x}}{2}.$$

In the following, Exercise 11.4.19 will be useful.

(a) Use Exercise 10.4.1 (p. 298) to deduce information about lists of such trees.

*(b) Use Exercise 10.4.1(c,d) to deduce information about sets and multisets of such trees. *Hint.* Show that

$$\exp\left(-\sqrt{1 - 4x}/2\right) \;=\; \frac{-\sqrt{1 - 4x}}{2}\sum_{k\ge 0}\frac{(1 - 4x)^k}{2^{2k}(2k + 1)!} + \sum_{k\ge 0}\frac{(1 - 4x)^k}{2^k(2k)!}.$$

11.4.21. Let D be a set of nonnegative integers containing 0. Let t_n be the number of rooted labeled n-vertex trees in which the outdegree of every vertex lies in D. Let $T(x)$ the EGF.

(a) Show that $T(x) \;=\; x\sum_{d\in D} T(x)^d/d!$.

*(b) Let $k = \gcd(D)$. Show that $t_n = 0$ when $n + 1$ is not a multiple of k.

(c) For finite D with $\gcd(D) = 1$, show how to estimate t_n.

*11.4.22. For each part of Exercise 11.2.2 except (c), discuss what sort of information about $\mathsf{E}_{\mathcal{T}}$ would be useful for estimating the coefficients of the exponential generating function given there. (See Exercise 11.4.19.)

Notes and References

The text by Sedgewick and Flajolet [15] covers some of the material in this chapter and Chapter 10, and also contains related material.

Further discussion of exponential generating functions can be found in many of the references given at the end of the previous chapter. Other generating functions besides ordinary and exponential ones play a role in mathematics. Dirichlet series play an important role in some aspects of analytic number theory. Apostol [1] gives an introduction to these series; however, some background in number theory or additional reading in his text is required. In combinatorics, almost all generating functions are ordinary or exponential. The next most important class, Gaussian generating functions, is associated with vector spaces over finite fields. Goldman and Rota [7] discuss them.

Lagrange inversion can be regarded as a theorem in complex analysis or as a theorem in combinatorics. In either case, it can be generalized to a set of simultaneous equations in several variables. Garsia and Shapiro [6] prove such a generalization combinatorially and give additional references. For readers familiar with complex analysis, here is a sketch of an analytic proof. Let $\sum a_n x^n = A(x) = g(T(x))$. By the Cauchy Residue Theorem followed by a change of variables,

$$na_n \;=\; \frac{1}{2\pi i}\oint \frac{A'(x)\,dx}{x^n} \;=\; \frac{1}{2\pi i}\oint \frac{g'(T(x))T'(x)\,dx}{x^n} \;=\; \frac{1}{2\pi i}\oint \frac{g'(T)\,dT}{(T/f(T))^n},$$

which equals the coefficient of u^{n-1} in $g'(u)f(u)^n$ by the Cauchy Residue Theorem.

Tutte's work on rooted maps (Exercise 11.2.18) was done in the 1960s. Connections with his work and the (asymptotic) enumeration of polyhedra are discussed in [3].

Pólya's theorem and some generalizations of it were first discovered by Redfield [14] whose paper was overlooked by mathematicians for about forty years. A translation of Pólya's paper together with some notes is available [13]. DeBruijn [4] give an excellent introduction to Pólya's theorem and some of its generalizations and applications. Harary and Palmer [9] discuss numerous applications in graph theory.

Textbooks on combinatorics generally avoid asymptotics. Wilf [16, Ch. 5] has a nice introduction to asymptotics which, in some ways, goes beyond ours. Books, such as the one by Greene and Knuth [8], that deal with analysis of algorithms may have some material on asymptotics. If you are interested in going beyond the material in this text, you should probably look at journal articles. The article by Bender [2] is an introduction to some methods, including ways to deal with the misfits in Example 11.32 (p. 353). (You should note that the hypotheses of Theorem 5 are too weak. A much more extensive discussion has been given by [12]. This has been corrected by Meir and Moon [11].) Our first principle for generating function asymptotics was adapted from the article by Flajolet and Odlyzko [5]. Pólya [13] discusses computing asymptotics for several classes of chemical compounds. A method for dealing with various types of trees, from combinatorial description to asymptotic formula, is discussed by Harary, Robinson and Schwenk [10].

1. Tom M. Apostol, *Introduction to Analytic Number Theory*, 5th ed., Springer-Verlag (1995).
2. Edward A. Bender, Asymptotic methods in enumeration, *SIAM Review* **16** (1974), 485–515. Errata: **18** (1976), 292.
3. Edward A. Bender, The number of three dimensional convex polyhedra. *American Math. Monthly* **94** (1987) 7–21.
4. Nicolas G. deBruijn, Pólya's theory of counting, 144–184 in E.F. Beckenbach (ed.), *Applied Combinatorial Mathematics*, John Wiley (1964).
5. Philippe Flajolet and Andrew Odlyzko, Singularity analysis of generating functions, *SIAM J. Discrete Math.* **3** (1990), 216–240.
6. Adriano M. Garsia and Joni Shapiro, Polynomials of binomial type and Lagrange inversion, *Proc. of the Amer. Math. Soc.* **64** (1977), 179–185.

7. Jay R. Goldman and Gian-Carlo Rota, On the foundations of combinatorial theory. IV. Finite vector spaces and Eulerian generating functions, *Studies in Applied Math.* **49** (1970), 239–258.
8. Daniel H. Greene and Donald E. Knuth, *Mathematics for the Analysis of Algorithms*, 3rd ed., Birkhäuser (1990).
9. Frank Harary and Edgar M. Palmer, *Graphical Enumeration*, Academic Press (1973).
10. Frank Harary, Robert W. Robinson and Allen J. Schwenk, Twenty-step algorithm for determining the asymptotic number of trees of various species, *J. Australian Math. Soc., Series A* **20** (1975), 483–503.
11. Amram Meir and John W. Moon, On an Asymptotic Method in Enumeration, *J. Combinatorial Theory, Series A* **51** (1989), 77–89.
12. Andrew M. Odlyzko, Asymptotic enumeration methods, 1063–1229 in R.L. Graham et al. (eds.), *Handbook of Combinatorics*, vol. 2, Elsevier (1995).
13. George Pólya and Ronald C. Read, *Combinatorial Enumeration of Groups, Graphs, and Chemical Compounds*, Springer-Verlag (1987). A translation with extensive notes of the article: George Pólya, Kombinatorische Anzahlbestimmungen für Gruppen, Graphen und chemische Verbindungen, *Acta Math.* **68** (1937), 145–254.
14. J.H. Redfield, The theory of group-reduced distributions, *American J. Math.* **49** (1927), 433-455.
15. Robert Sedgwick and Philippe Flajolet, *An Introduction to the Analysis of Algorithms*, Addison-Wesley (1996).
16. Herbert S. Wilf, *Generatingfunctionology*, 2nd ed., Academic Press (1994).