Chapter 8

Recursion and Recurrence Relations

Fibonacci sequence

Zero, one! One, two, three! Five and eight!
Then thirteen, twenty-one! At this rate
Fibonacci appears;
The man's sequence for years
Has kept math students studying late.

Goldie, The Omnificent English Dictionary In Limerick Form

An essential tool that anyone interested in computer science must master is how to think recursively. The ability to understand definitions, concepts, algorithms, etc., that are presented recursively and the ability to put thoughts into a recursive framework are essential in computer science. One of our goals in this chapter is to help the reader become more comfortable with recursion in its commonly encountered forms.

A second goal is to discuss recurrence relations. We will concentrate on methods of solving recurrence relations, including an introduction to generating functions.

8.1 The Many Faces of Recursion

Consider the following definitions, all of which should be somewhat familiar to you. When reading them, concentrate on how they are similar.

8.1.1 Binomial Coefficients

Here is a recursive definition of binomial coefficients, which we introduced in Chapter 2.

Definition 8.1.1 Binomial Coefficient - Recursion Definition. Assume $n \geq 0$ and $n \geq k \geq 0$. We define $\binom{n}{k}$ by

- $\displaystyle\binom{n}{0} = 1$
- $\binom{n}{n} = 1$ and

- $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$ if $n > k > 0$

♢

Observation 8.1.2 A word about definitions: Strictly speaking, when mathematical objects such as binomial coefficients are defined, they should be defined just once. Since we defined binomial coefficients earlier, in Definition 2.4.3, other statements describing them should be theorems. The theorem, in this case, would be that the "definition" above is consistent with the original definition. Our point in this chapter in discussing recursion is to observe alternative definitions that have a recursive nature. In the exercises, you will have the opportunity to prove that the two definitions are indeed equivalent.

Here is how we can apply the recursive definition to compute $\binom{5}{2}$.

$$\begin{aligned}
\binom{5}{2} &= \binom{4}{2} + \binom{4}{1} \\
&= (\binom{3}{2} + \binom{3}{1}) + (\binom{3}{1} + \binom{3}{0}) \\
&= \binom{3}{2} + 2\binom{3}{1} + 1 \\
&= (\binom{2}{2} + \binom{2}{1}) + 2(\binom{2}{1} + \binom{2}{0}) + 1 \\
&= (1 + \binom{2}{1}) + 2(\binom{2}{1} + 1) + 1 \\
&= 3\binom{2}{1} + 4 \\
&= 3(\binom{1}{1} + \binom{1}{0}) + 4 \\
&= 3(1+1) + 4 = 10
\end{aligned}$$

8.1.2 Polynomials and Their Evaluation

Definition 8.1.3 Polynomial Expression in x over S (Non-Recursive). Let n be an integer, $n \geq 0$. An n^{th} degree polynomial in x is an expression of the form $a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$, where $a_n, a_{n-1}, \ldots, a_1, a_0$ are elements of some designated set of numbers, S, called the set of coefficients and $a_n \neq 0$. ♢

We refer to x as a variable here, although the more precise term for x is an *indeterminate*. There is a distinction between the terms indeterminate and variable, but that distinction will not come into play in our discussions.

Zeroth degree polynomials are called constant polynomials and are simply elements of the set of coefficients.

This definition is often introduced in algebra courses to describe expressions such as $f(n) = 4n^3 + 2n^2 - 8n + 9$, a third-degree, or cubic, polynomial in n. This definition has a drawback when the variable is given a value and the expression must be evaluated. For example, suppose that $n = 7$. Your first impulse is likely to do this:

$$\begin{aligned}
f(7) &= 4 \cdot 7^3 + 2 \cdot 7^2 - 8 \cdot 7 + 9 \\
&= 4 \cdot 343 + 2 \cdot 49 - 8 \cdot 7 + 9 \\
&= 1423
\end{aligned}$$

A count of the number of operations performed shows that five multiplications and three additions/subtractions were performed. The first two multiplications compute 7^2 and 7^3, and the last three multiply the powers of 7 times the coefficients. This gives you the four terms; and adding/subtracting a list of k numbers requires $k-1$ addition/subtractions. The following definition of a polynomial expression suggests another more efficient method of evaluation.

Definition 8.1.4 Polynomial Expression in x over S (Recursive). Let S be a set of coefficients and x a variable.

(a) A zeroth degree polynomial expression in x over S is a nonzero element of S.

(b) For $n \geq 1$, an n^{th} degree polynomial expression in x over S is an expression of the form $p(x)x+a$ where $p(x)$ is an $(n-1)^{st}$ degree polynomial expression in x and $a \in S$.

◊

We can easily verify that $f(n) = 4n^3 + 2n^2 - 8n + 9$ is a third-degree polynomial expression in n over $\mathbb{Z}$ based on this definition:

$$f(n) = 4n^3 + 2n^2 - 8n + 9 = ((4n+2)n - 8)n + 9$$

Notice that 4 is a zeroth degree polynomial since it is an integer. Therefore $4n+2$ is a first-degree polynomial; therefore, $(4n+2)n-8$ is a second-degree polynomial in n over $\mathbb{Z}$; therefore, $f(n)$ is a third-degree polynomial in n over $\mathbb{Z}$. The final expression for $f(n)$ is called its **telescoping form**. If we use it to calculate $f(7)$, we need only three multiplications and three additions/subtractions. This is called **Horner's method** for evaluating a polynomial expression.

Example 8.1.5 More Telescoping Polynomials.

(a) The telescoping form of $p(x) = 5x^4 + 12x^3 - 6x^2 + x + 6$ is $(((5x+12)x-6)x+1)x+6$. Using Horner's method, computing the value of $p(c)$ requires four multiplications and four additions/subtractions for any real number c.

(b) $g(x) = -x^5 + 3x^4 + 2x^2 + x$ has the telescoping form $((((-x+3)x)x+2)x+1)x$.

□

Many computer languages represent polynomials as lists of coefficients, usually starting with the constant term. For example, $g(x) = -x^5 + 3x^4 + 2x^2 + x$ would be represented with the list $\{0, 1, 2, 0, 3, -1\}$. In both Mathematica and Sage, polynomial expressions can be entered and manipulated, so the list representation is only internal. Some programming languages require users to program polynomial operations with lists. We will leave these programming issues to another source.

8.1.3 Recursive Searching - The Binary Search

Next, we consider a recursive algorithm for a binary search within a sorted list of items. Suppose $r = \{r(1), r(2), \ldots, r(n)\}$ represent a list of n items sorted by a numeric key in descending order. The j^{th} item is denoted $r(j)$ and its key value by $r(j)$.`key`. For example, each item might contain data on the buildings in a city and the key value might be the height of the building. Then $r(1)$ would be the item for the tallest building and $r(1)$.`key` would be its height. The algorithm `BinarySearch`(j, k) can be applied to search for an item in r with key value C. This would be accomplished by the execution of `BinarySearch`$(1, n)$. When the algorithm is completed, the variable `Found` will have a value of `true` if an item with the desired key value was found, and the value of location will be the index of an item whose key is C. If `Found` keeps the value `false`, no such item exists in the list. The general idea behind the algorithm is illustrated in Figure 8.1.6

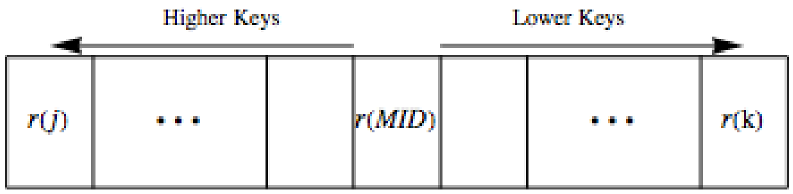

Figure 8.1.6 General Scheme for a Binary Search

In the following implementation of the Binary Search in SageMath, we search within a sorted list of integers. Therefore, the items themselves are the keys.

```
def BinarySearch(r,j,k,C):
   found = False
   if j <= k:
      mid = floor((j + k)/2)
      print('probing␣at␣position␣'+str(mid))
      if r[mid] == C:
         location = mid
         found = True
         print('found␣in␣position␣'+str(location))
         return location
      else:
        if r[mid] > C:
           BinarySearch(r,j, mid - 1,C)
        else:
           BinarySearch(r,mid + 1,k,C)
   else:
      print('not␣found')
      return False
s=[1,9,13,16,30,31,32,33,36,37,38,45,49,50,52,61,63,64,69,77,79,80,81,83,86,90,93,96]
BinarySearch(s,0,len(s)-1,30)
```

```
probing at position 13
probing at position 6
probing at position 2
probing at position 4
found in position 4
```

8.1.4 Recursively Defined Sequences

For the next two examples, consider a sequence of numbers to be a list of numbers consisting of a zeroth number, first number, second number, ... If a sequence is given the name S, the k^{th} number of S is usually written S_k or $S(k)$.

Example 8.1.7 Geometric Growth Sequence. Define the sequence of numbers B by

$$B_0 = 100 \text{ and}$$

$$B_k = 1.08B_{k-1} \text{ for } k \geq 1.$$

These rules stipulate that each number in the list is 1.08 times the previous number, with the starting number equal to 100. For example

$$\begin{aligned} B_3 &= 1.08B_2 \\ &= 1.08\,(1.08B_1) \\ &= 1.08\,(1.08\,(1.08B_0)) \\ &= 1.08(1.08(1.08 \cdot 100)) \\ &= 1.08^3 \cdot 100 = 125.971 \end{aligned}$$

□

Example 8.1.8 The Fibonacci Sequence. The Fibonacci sequence is the sequence F defined by

$$F_0 = 1,\ F_1 = 1 \text{ and}$$

$$F_k = F_{k-2} + F_{k-1} \text{ for } k \geq 2$$

□

8.1.5 Recursion

All of the previous examples were presented recursively. That is, every "object" is described in one of two forms. One form is by a simple definition, which is usually called the basis for the recursion. The second form is by a recursive description in which objects are described in terms of themselves, with the following qualification. What is essential for a proper use of recursion is that the objects can be expressed in terms of simpler objects, where "simpler" means closer to the basis of the recursion. To avoid what might be considered a circular definition, the basis must be reached after a finite number of applications of the recursion.

To determine, for example, the fourth item in the Fibonacci sequence we repeatedly apply the recursive rule for F until we are left with an expression involving F_0 and F_1:

$$\begin{aligned} F_4 &= F_2 + F_3 \\ &= (F_0 + F_1) + (F_1 + F_2) \\ &= (F_0 + F_1) + (F_1 + (F_0 + F_1)) \\ &= (1+1) + (1 + (1+1)) \\ &= 5 \end{aligned}$$

8.1.6 Iteration

On the other hand, we could compute a term in the Fibonacci sequence such as F_5 by starting with the basis terms and working forward as follows:

Table 8.1.9

$$\begin{aligned} F_2 &= F_0 + F_1 = 1 + 1 = 2 \\ F_3 &= F_1 + F_2 = 1 + 2 = 3 \\ F_4 &= F_2 + F_3 = 2 + 3 = 5 \\ F_5 &= F_3 + F_4 = 3 + 5 = 8 \end{aligned}$$

This is called an iterative computation of the Fibonacci sequence. Here we start with the basis and work our way forward to a less simple number, such as 5. Try to compute F_5 using the recursive definition for F as we did for F_4. It will take much more time than it would have taken to do the computations above. Iterative computations usually tend to be faster than computations that apply recursion. Therefore, one useful skill is being able to convert a recursive formula into a nonrecursive formula, such as one that requires only iteration or a faster method, if possible.

An iterative formula for $\binom{n}{k}$ is also much more efficient than an application of the recursive definition. The recursive definition is not without its merits, however. First, the recursive equation is often useful in manipulating algebraic expressions involving binomial coefficients. Second, it gives us an insight into the combinatoric interpretation of $\binom{n}{k}$. In choosing k elements from $\{1, 2, ..., n\}$, there are $\binom{n-1}{k}$ ways of choosing all k from $\{1, 2, ..., n-1\}$, and there are $\binom{n-1}{k-1}$ ways of choosing the k elements if n is to be selected and the remaining $k-1$ elements come from $\{1, 2, ..., n-1\}$. Note how we used the Law of Addition from Chapter 2 in our reasoning.

BinarySearch Revisited. In the binary search algorithm, the place where recursion is used is easy to pick out. When an item is examined and the key is not the one you want, the search is cut down to a sublist of no more than half the number of items that you were searching in before. Obviously, this is a simpler search. The basis is hidden in the algorithm. The two cases that complete the search can be thought of as the basis. Either you find an item that you want, or the sublist that you have been left to search in is empty, when $j > k$.

BinarySearch can be translated without much difficulty into any language that allows recursive calls to its subprograms. The advantage to such a program is that its coding would be much shorter than a nonrecursive program that does a binary search. However, in most cases the recursive version will be slower and require more memory at execution time.

8.1.7 Induction and Recursion

The definition of the positive integers in terms of Peano's Postulates is a recursive definition. The basis element is the number 1 and the recursion is that if n is a positive integer, then so is its successor. In this case, n is the simple object and the recursion is of a forward type. Of course, the validity of an induction proof is based on our acceptance of this definition. Therefore, the appearance of induction proofs when recursion is used is no coincidence.

Example 8.1.10 Proof of a formula for B. A formula for the sequence B in Example 8.1.7 is $B = 100(1.08)^k$ for $k \geq 0$. A proof by induction follow.

If $k = 0$, then $B = 100(1.08)^0 = 100$, as defined. Now assume that for some $k \geq 1$, the formula for B_k is true.

$$\begin{aligned} B_{k+1} &= 1.08 B_k \text{ by the recursive definition} \\ &= 1.08 \left(100(1.08)^k\right) \text{ by the induction hypothesis} \\ &= 100(1.08)^{k+1} \end{aligned}$$

hence the formula is true for $k + 1$

The formula that we have just proven for B is called a closed form expression. It involves no recursion or summation signs. □

Definition 8.1.11 Closed Form Expression. Let $E = E(x_1, x_2, \ldots, x_n)$ be an algebraic expression involving variables $x_1, x_2, \ldots, x_n$ which are allowed to take on values from some predetermined set. E is a **closed form expression** if there exists a number T such that the evaluation of E with any allowed values of the variables will take no more than T operations (alternatively, T time units). ♢

Example 8.1.12 Reducing a summation to closed form. The sum $E(n) = \sum_{k=1}^{n} k$ is not a closed form expression because the number of additions needed to evaluate $E(n)$ grows indefinitely with n. A closed form expression that computes the value of $E(n)$ is $\frac{n(n+1)}{2}$, which only requires $T = 3$ operations. □

8.1.8 Exercises

1. By the recursive definition of binomial coefficients, $\binom{7}{2} = \binom{6}{2} + \binom{6}{1}$. Continue expanding $\binom{7}{2}$ to express it in terms of quantities defined by the basis. Check your result by applying the factorial definition of $\binom{n}{k}$.

2. Define the sequence L by $L_0 = 5$ and for $k \geq 1$, $L_k = 2L_{k-1} - 7$. Determine L_4 and prove by induction that $L_k = 7 - 2^{k+1}$.

3. Let $p(x) = x^5 + 3x^4 - 15x^3 + x - 10$.

(a) Write $p(x)$ in telescoping form.

(b) Use a calculator to compute $p(3)$ using the original form of $p(x)$.

(c) Use a calculator to compute $p(3)$ using the telescoping form of $p(x)$.

(d) Compare your speed in parts b and c.

4. Suppose that a list of nine items, $(r(l), r(2), ..., r(9))$, is sorted by key in decending order so that $r(3).\texttt{key} = 12$ and $r(4).\texttt{key} = 10$. List the executions of the BinarySearch algorithms that would be needed to complete BinarySearch(1,9) when:

(a) The search key is C = 12 (b) The search key is C = 11

Assume that distinct items have distinct keys.

5. What is wrong with the following definition of $f : \mathbb{R} \to \mathbb{R}$? $f(0) = 1$ and $f(x) = f(x/2)/2$ if $x \neq 0$.

6. Prove the two definitions of binomials coefficients, Definition 2.4.3 and Definition 8.1.1, are equivalent.

7. Prove by induction that if $n \geq 0$, $\sum_{k=0}^{n} \binom{n}{k} = 2^n$

8.2 Sequences

8.2.1 Sequences and Ways They Are Defined

Definition 8.2.1 Sequence. A sequence is a function from the natural numbers into some predetermined set. The image of any natural number k can be written as $S(k)$ or S_k and is called the k^{th} *term* of S. The variable k is called the *index* or *argument* of the sequence. ♢

For example, a sequence of integers would be a function $S : \mathbb{N} \to \mathbb{Z}$.

Example 8.2.2 Three sequences defined in different ways.

(a) The sequence A defined by $A(k) = k^2 - k$, $k \geq 0$, is a sequence of integers.

(b) The sequence B defined recursively by $B(0) = 2$ and $B(k) = B(k-1) + 3$ for $k \geq 1$ is a sequence of integers. The terms of B can be computed either by applying the recursion formula or by iteration. For example,

$$\begin{aligned}B(3) &= B(2)+3\\ &= (B(1)+3)+3\\ &= ((B(0)+3)+3)+3\\ &= ((2+3)+3)+3 = 11\end{aligned}$$

or

$$\begin{aligned}B(1) = B(0)+3 = 2+3 = 5\\ B(2) = B(1)+3 = 5+3 = 8\\ B(3) = B(2)+3 = 8+3 = 11\end{aligned}$$

(c) Let C_r be the number of strings of 0's and 1's of length r having no consecutive zeros. These terms define a sequence C of integers.

□

Remarks:

(1) A sequence is often called a *discrete function*.

(2) Although it is important to keep in mind that a sequence is a function, another useful way of visualizing a sequence is as a list. For example, the sequence A in the previous example could be written as $(0, 0, 2, 6, 12, 20, \dots)$. Finite sequences can appear much the same way when they are the input to or output from a computer. The index of a sequence can be thought of as a time variable. Imagine the terms of a sequence flashing on a screen every second. Then s_k would be what you see in the k^{th} second. It is convenient to use terminology like this in describing sequences. For example, the terms that precede the k^{th} term of A would be $A(0), A(1), ..., A(k-1)$. They might be called the earlier terms.

8.2.2 A Fundamental Problem

Given the definition of any sequence, a fundamental problem that we will concern ourselves with is to devise a method for determining any specific term in a minimum amount of time. Generally, time can be equated with the number of operations needed. In counting operations, the application of a recursive formula would be considered an operation.

(a) The terms of A in Example 8.2.2 are very easy to compute because of the closed form expression. No matter what term you decide to compute, only three operations need to be performed.

(b) How to compute the terms of B is not so clear. Suppose that you wanted to know $B(100)$. One approach would be to apply the definition recursively:

$$B(100) = B(99)+3 = (B(98)+3)+3 = \cdots$$

The recursion equation for B would be applied 100 times and 100 additions would then follow. To compute $B(k)$ by this method, $2k$ operations are needed. An iterative computation of $B(k)$ is an improvement: $B(1) = B(0)+3 = 2+3 = 5$

$B(2) = B(1)+3 = 5+3 = 8$

etc. Only k additions are needed. This still isn't a good situation. As k gets large, we take more and more time to compute $B(k)$. The formula

$B(k) = B(k-1) + 3$ is called a recurrence relation on B. The process of finding a closed form expression for $B(k)$, one that requires no more than some fixed number of operations, is called solving the recurrence relation.

(c) The determination of C_k is a standard kind of problem in combinatorics. One solution is by way of a recurrence relation. In fact, many problems in combinatorics are most easily solved by first searching for a recurrence relation and then solving it. The following observation will suggest the recurrence relation that we need to determine C_k. If $k \geq 2$, then every string of 0's and 1's with length k and no two consecutive 0's is either $1s_{k-1}$ or $01s_{k-2}$, where s_{k-1} and s_{k-2} are strings with no two consecutive 0's of length $k-1$ and $k-2$ respectively. From this observation we can see that $C_k = C_{k-2} + C_{k-1}$ for $k \geq 2$. The terms $C_0 = 1$ and $C_1 = 2$ are easy to determine by enumeration. Now, by iteration, any C_k can be easily determined. For example, $C_5 = 21$ can be computed with five additions. A closed form expression for C_k would be an improvement. Note that the recurrence relation for C_k is identical to the one for The Fibonacci Sequence. Only the basis is different.

8.2.3 Exercises

1. Prove by induction that $B(k) = 3k + 2$, $k \geq 0$, is a closed form expression for the sequence B in Example 8.2.2

2.

(a) Consider sequence Q defined by $Q(k) = 2k + 9$, $k \geq 1$. Complete the table below and determine a recurrence relation that describes Q.

k	$Q(k)$	$Q(k) - Q(k-1)$
2		
3		
4		
5		
6		
7		

(b) Let $A(k) = k^2 - k$, $k \geq 0$. Complete the table below and determine a recurrence relation for A.

k	$A(k)$	$A(k) - A(k-1)$	$A(k) - 2A(k-1) + A(k-2)$
2			
3			
4			
5			

3. Given k lines ($k \geq 0$) on a plane such that no two lines are parallel and no three lines meet at the same point, let $P(k)$ be the number of regions into which the lines divide the plane (including the infinite ones (see Figure 8.2.3). Describe how the recurrence relation $P(k) = P(k-1) + k$ can be derived. Given that $P(0) = 1$, determine $P(5)$.

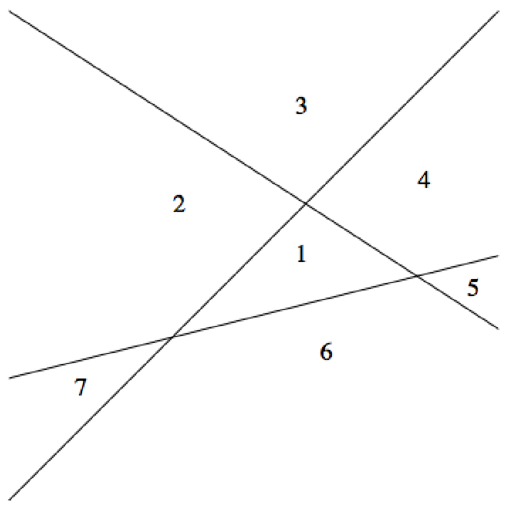

Figure 8.2.3 A general configuration of three lines

4. A sample of a radioactive substance is expected to decay by 0.15 percent each hour. If w_t, $t \geq 0$, is the weight of the sample t hours into an experiment, write a recurrence relation for w.

5. Let $M(n)$ be the number of multiplications needed to evaluate an n^{th} degree polynomial. Use the recursive definition of a polynomial expression to define M recursively.

6. Let S be sequence of integers. Using short English sentences, not symbols, describe what the following propositions say about S. Are the two propositions equivalent?

(a) $(\forall M)_{\mathbb{N}}((\exists n)_{\mathbb{N}}(S(n) \geq M))$

(b) $(\forall M)_{\mathbb{N}}((\exists N)_{\mathbb{N}}((\forall n)_{\mathbb{N}}(n \geq N \rightarrow S(n) \geq M)))$

8.3 Recurrence Relations

In this section we will begin our study of recurrence relations and their solutions. Our primary focus will be on the class of finite order linear recurrence relations with constant coefficients (shortened to finite order linear relations). First, we will examine closed form expressions from which these relations arise. Second, we will present an algorithm for solving them. In later sections we will consider some other common relations (8.4) and introduce two additional tools for studying recurrence relations: generating functions (8.5) and matrix methods (Chapter 12).

8.3.1 Definition and Terminology

Definition 8.3.1 Recurrence Relation. Let S be a sequence of numbers. A recurrence relation on S is a formula that relates all but a finite number of terms of S to previous terms of S. That is, there is a k_0 in the domain of S such that if $k \geq k_0$, then $S(k)$ is expressed in terms of some (and possibly all) of the terms that precede $S(k)$. If the domain of S is $\{0, 1, 2, \ldots\}$, the terms $S(0), S(1), ..., S(k_0 - 1)$ are not defined by the recurrence formula. Their values are the initial conditions (or boundary conditions, or basis) that complete the definition of S. ◊

Example 8.3.2 Some Examples of Recurrence Relations.

(a) The Fibonacci sequence is defined by the recurrence relation $F_k = F_{k-2} + F_{k-1}$, $k \geq 2$, with the initial conditions $F_0 = 1$ and $F_1 = 1$. The recurrence relation is called a second-order relation because F_k depends on the two previous terms of F. Recall that the sequence C in Section 8.2, Example 8.2.2, can be defined with the same recurrence relation, but with different initial conditions.

(b) The relation $T(k) = 2T(k-1)^2 - kT(k-3)$ is a third-order recurrence relation. If values of $T(0)$, $T(1)$, and $T(2)$ are specified, then T is completely defined.

(c) The recurrence relation $S(n) = S(\lfloor n/2 \rfloor) + 5$, $n > 0$, with $S(0) = 0$ has infinite order. To determine $S(n)$ when n is even, you must go back $n/2$ terms. Since $n/2$ grows unbounded with n, no finite order can be given to S.

□

8.3.2 Solving Recurrence Relations

Sequences are often most easily defined with a recurrence relation; however, the calculation of terms by directly applying a recurrence relation can be time-consuming. The process of determining a closed form expression for the terms of a sequence from its recurrence relation is called solving the relation. There is no single technique or algorithm that can be used to solve all recurrence relations. In fact, some recurrence relations cannot be solved. The relation that defines T above is one such example. Most of the recurrence relations that you are likely to encounter in the future are classified as finite order linear recurrence relations with constant coefficients. This class is the one that we will spend most of our time with in this chapter.

Definition 8.3.3 n^{th} Order Linear Recurrence Relation. Let S be a sequence of numbers with domain $k \geq 0$. An n^{th} order linear recurrence relation on S with constant coefficients is a recurrence relation that can be written in the form

$$S(k) + C_1 S(k-1) + \ldots + C_n S(k-n) = f(k) \text{ for } k \geq n$$

where $C_1, C_2, \ldots, C_n$ are constants and f is a numeric function that is defined for $k \geq n$. ♢

Note: We will shorten the name of this class of relations to n^{th} order linear relations. Therefore, in further discussions, $S(k) + 2kS(k-1) = 0$ would not be considered a first-order linear relation.

Example 8.3.4 Some Finite Order Linear Relations.

(a) The Fibonacci sequence is defined by the second-order linear relation because $F_k - F_{k-1} - F_{k-2} = 0$

(b) The relation $P(j) + 2P(j-3) = j^2$ is a third-order linear relation. In this case, $C_1 = C_2 = 0$.

(c) The relation $A(k) = 2(A(k-1) + k)$ can be written as $A(k) - 2A(k-1) = 2k$. Therefore, it is a first-order linear relation.

□

8.3.3 Recurrence Relations Obtained from "Solutions"

Before giving an algorithm for solving finite order linear relations, we will examine recurrence relations that arise from certain closed form expressions. The closed form expressions are selected so that we will obtain finite order linear relations from them. This approach may seem a bit contrived, but if you were to write down a few simple algebraic expressions, chances are that most of them would be similar to the ones we are about to examine.

For our first example, consider D, defined by $D(k) = 5 \cdot 2^k$, $k \geq 0$. If $k \geq 1$, $D(k) = 5 \cdot 2^k = 2 \cdot 5 \cdot 2^{k-1} = 2D(k-1)$. Therefore, D satisfies the first order linear relation $D(k) - 2D(k-1) = 0$ and the initial condition $D(0) = 5$ serves as an initial condition for D.

As a second example, consider $C(k) = 3^{k-1} + 2^{k+1} + k$, $k \geq 0$. Quite a bit more algebraic manipulation is required to get our result:

Table 8.3.5

$C(k) = 3^{k-1} + 2^{k+1} + k$	Original equation
$3C(k-1) = 3^{k-1} + 3 \cdot 2^k + 3(k-1)$	Substitute $k-1$ for k and multiply by 3
	Subtract the second equation from the first.
$C(k) - 3C(k-1) = -2^k - 2k + 3$	3^{k-1} term is eliminated. This is a first order relation.
$2C(k-1) - 6C(k-2) = -2^k - 2(2(k-1)) + 6$	Substitute $k-1$ for k in the third equation, multiply by 2.
	Subtract the 4th equation from the 3rd.
$C(k) - 5C(k-1) + 6C(k-2) = 2k - 7$	2^{k+1}term is eliminated. This is 2nd order relation.

The recurrence relation that we have just obtained, defined for $k \geq 2$, together with the initial conditions $C(0) = 7/3$ and $C(1) = 6$, define C.

Table 8.3.6 summarizes our results together with a few other examples that we will let the reader derive. Based on these results, we might conjecture that any closed form expression for a sequence that combines exponential expressions and polynomial expressions will be solutions of finite order linear relations. Not only is this true, but the converse is true: a finite order linear relation defines a closed form expression that is similar to the ones that were just examined. The only additional information that is needed is a set of initial conditions.

Table 8.3.6 Recurrence Relations Obtained from Given Sequences

Closed Form Expression	Recurrence Relation
$D(k) = 5 \cdot 2^k$	$D(k) - 2D(k-1) = 0$
$C(k) = 3^{k-1} + 2^{k+1} + k$	$C(k) - 2C(k-1) - 6C(k-2) = 2k - 7$
$Q(k) = 2k + 9$	$Q(k) - Q(k-1) = 2$
$A(k) = k^2 - k$	$A(k) - 2A(k-1) + A(k-2) = 2$
$B(k) = 2k^2 + 1$	$B(k) - 2B(k-1) + B(k-2) = 4$
$G(k) = 2 \cdot 4^k - 5(-3)^k$	$G(k) - G(k-1) + 12G(k-2) = 0$
$J(k) = (3+k)2^k$	$J(k) - 4J(k-1) + 4J(k-2) = 0$

Definition 8.3.7 Homogeneous Recurrence Relation. An n^{th} order linear relation is homogeneous if $f(k) = 0$ for all k. For each recurrence relation $S(k) + C_1 S(k-1) + \ldots + C_n S(k-n) = f(k)$, the associated homogeneous relation is $S(k) + C_1 S(k-1) + \ldots + C_n S(k-n) = 0$ ♢

Example 8.3.8 First Order Homogeneous Recurrence Relations. $D(k) - 2D(k-1) = 0$ is a first-order homogeneous relation. Since it can also be written as $D(k) = 2D(k-1)$, it should be no surprise that it arose from an expression that involves powers of 2. More generally, you would expect that the solution of $L(k) - aL(k-1)$ would involve a^k. Actually, the solution is $L(k) = L(0)a^k$, where the value of $L(0)$ is given by the initial condition. □

Example 8.3.9 A Second Order Example. Consider the second-order homogeneous relation $S(k) - 7S(k-1) + 12S(k-2) = 0$ together with the initial conditions $S(0) = 4$ and $S(1) = 4$. From our discussion above, we can predict that the solution to this relation involves terms of the form ba^k, where b and a are nonzero constants that must be determined. If the solution were to equal this quantity exactly, then

$$\begin{aligned} S(k) &= ba^k \\ S(k-1) &= ba^{k-1} \\ S(k-2) &= ba^{k-2} \end{aligned}$$

Substitute these expressions into the recurrence relation to get

$$ba^k - 7ba^{k-1} + 12ba^{k-2} = 0$$

Each term on the left-hand side of this equation has a factor of ba^{k-2}, which is nonzero. Dividing through by this common factor yields

$$a^2 - 7a + 12 = (a-3)(a-4) = 0 \tag{8.3.1}$$

Therefore, the only possible values of a are 3 and 4. Equation (8.3.1) is called the characteristic equation of the recurrence relation. The fact is that our original recurrence relation is true for any sequence of the form $S(k) = b_1 3^k + b_2 4^k$, where b_1 and b_2 are real numbers. This set of sequences is called the general solution of the recurrence relation. If we didn't have initial conditions for S, we would stop here. The initial conditions make it possible for us to find definite values for b_1 and b_2.

$$\left\{ \begin{array}{c} S(0) = 4 \\ S(1) = 4 \end{array} \right\} \Rightarrow \left\{ \begin{array}{c} b_1 3^0 + b_2 4^0 = 4 \\ b_1 3^1 + b_2 4^1 = 4 \end{array} \right\} \Rightarrow \left\{ \begin{array}{c} b_1 + b_2 = 4 \\ 3b_1 + 4b_2 = 4 \end{array} \right\}$$

The solution of this set of simultaneous equations is $b_1 = 12$ and $b_2 = -8$ and so the solution is $S(k) = 12 \cdot 3^k - 8 \cdot 4^k$. □

Definition 8.3.10 Characteristic Equation. The characteristic equation of the homogeneous n^{th} order linear relation $S(k) + C_1 S(k-1) + \ldots + C_n S(k-n) = 0$ is the nth degree polynomial equation

$$a^n + \sum_{j=1}^{n} C_j a^{n-j} = a^n + C_1 a^{n-1} + \cdots + C_{n-1} a + C_n = 0$$

The left-hand side of this equation is called the characteristic polynomial. The roots of the characteristic polynomial are called the characteristic roots of the equation. ◊

Example 8.3.11 Some characteristic equations.

(a) The characteristic equation of $F(k) - F(k-1) - F(k-2) = 0$ is $a^2 - a - 1 = 0$.

(b) The characteristic equation of $Q(k) + 2Q(k-1) - 3Q(k-2) - 6Q(k-4) = 0$

is $a^4 + 2a^3 - 3a^2 - 6 = 0$. Note that the absence of a $Q(k-3)$ term means that there is not an $x^{4-3} = x$ term appearing in the characteristic equation.

□

Algorithm 8.3.12 Algorithm for Solving Homogeneous Finite Order Linear Relations.

(a) *Write out the characteristic equation of the relation* $S(k) + C_1 S(k-1) + \ldots + C_n S(k-n) = 0$, *which is* $a^n + C_1 a^{n-1} + \cdots + C_{n-1} a + C_n = 0$.

(b) *Find all roots of the characteristic equation, the characteristic roots.*

(c) *If there are* n *distinct characteristic roots,* $a_1, a_2, \ldots a_n$, *then the general solution of the recurrence relation is* $S(k) = b_1 {a_1}^k + b_2 {a_2}^k + \cdots + b_n {a_n}^k$. *If there are fewer than* n *characteristic roots, then at least one root is a multiple root. If* a_j *is a double root, then the* $b_j {a_j}^k$ *term is replaced with* $(b_{j,0} + b_{j,1} k)\, a_j^k$. *In general, if* a_j *is a root of multiplicity* p, *then the* $b_j {a_j}^k$ *term is replaced with* $\left(b_{j,0} + b_{j,1} k + \cdots + b_{j,(p-1)} k^{p-1}\right) a_j^k$.

(d) *If* n *initial conditions are given, we get* n *linear equations in* n *unknowns (the* ${b_j}'s$ *from Step 3) by substitution. If possible, solve these equations to determine a final form for* $S(k)$.

Although this algorithm is valid for all values of n, there are limits to the size of n for which the algorithm is feasible. Using just a pencil and paper, we can always solve second-order equations. The quadratic formula for the roots of $ax^2 + bx + c = 0$ is

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

The solutions of $a^2 + C_1 a + C_2 = 0$ are then

$$\frac{1}{2}\left(-C_1 + \sqrt{{C_1}^2 - 4C_2}\right) \text{ and } \frac{1}{2}\left(-C_1 - \sqrt{{C_1}^2 - 4C_2}\right)$$

Although cubic and quartic formulas exist, they are too lengthy to introduce here. For this reason, the only higher-order relations ($n \geq 3$) that you could be expected to solve by hand are ones for which there is an easy factorization of the characteristic polynomial.

Example 8.3.13 A solution using the algorithm. Suppose that T is defined by $T(k) = 7T(k-1) - 10T(k-2)$, with $T(0) = 4$ and $T(1) = 17$. We can solve this recurrence relation with Algorithm 8.3.12:

(a) Note that we have written the recurrence relation in "nonstandard" form. To avoid errors in this easy step, you might consider a rearrangement of the equation to, in this case, $T(k) - 7T(k-1) + 10T(k-2) = 0$. Therefore, the characteristic equation is $a^2 - 7a + 10 = 0$.

(b) The characteristic roots are $\frac{1}{2}\left(7 + \sqrt{49 - 40}\right) = 5$ and $\frac{1}{2}\left(7 - \sqrt{49 - 40}\right) = 2$. These roots can be just as easily obtained by factoring the characteristic polynomial into $(a-5)(a-2)$.

(c) The general solution of the recurrence relation is $T(k) = b_1 2^k + b_2 5^k$.

(d) $$\left\{\begin{array}{c} T(0) = 4 \\ T(1) = 17 \end{array}\right\} \Rightarrow \left\{\begin{array}{c} b_1 2^0 + b_2 5^0 = 4 \\ b_1 2^1 + b_2 5^1 = 17 \end{array}\right\} \Rightarrow \left\{\begin{array}{c} b_1 + b_2 = 4 \\ 2b_1 + 5b_2 = 17 \end{array}\right\}$$
The simultaneous equations have the solution $b_1 = 1$ and $b_2 = 3$. Therefore, $T(k) = 2^k + 3 \cdot 5^k$.

□

Here is one rule that might come in handy: If the coefficients of the characteristic polynomial are all integers, with the constant term equal to m, then the only possible rational characteristic roots are divisors of m (both positive and negative).

With the aid of a computer (or possibly only a calculator), we can increase n. Approximations of the characteristic roots can be obtained by any of several well-known methods, some of which are part of standard software packages. There is no general rule that specifies the values of n for which numerical approximations will be feasible. The accuracy that you get will depend on the relation that you try to solve. (See Exercise 17 of this section.)

Example 8.3.14 Solution of a Third Order Recurrence Relation. Solve $S(k) - 7S(k-2) + 6S(k-3) = 0$, where $S(0) = 8$, $S(1) = 6$, and $S(2) = 22$.

(a) The characteristic equation is $a^3 - 7a + 6 = 0$.

(b) The only rational roots that we can attempt are $\pm 1, \pm 2, \pm 3, \text{and} \pm 6$. By checking these, we obtain the three roots 1, 2, and -3.

(c) The general solution is $S(k) = b_1 1^k + b_2 2^k + b_3(-3)^k$. The first term can simply be written b_1 .

(d) $\left\{ \begin{array}{c} S(0) = 8 \\ S(1) = 6 \\ S(2) = 22 \end{array} \right\} \Rightarrow \left\{ \begin{array}{c} b_1 + b_2 + b_3 = 8 \\ b_1 + 2b_2 - 3b_3 = 6 \\ b_1 + 4b_2 + 9b_3 = 22 \end{array} \right\}$ You can solve this system by elimination to obtain $b_1 = 5$, $b_2 = 2$, and $b_3 = 1$. Therefore, $S(k) = 5 + 2 \cdot 2^k + (-3)^k = 5 + 2^{k+1} + (-3)^k$

□

Example 8.3.15 Solution with a Double Characteristic Root. Solve $D(k) - 8D(k-1) + 16D(k-2) = 0$, where $D(2) = 16$ and $D(3) = 80$.

(a) Characteristic equation: $a^2 - 8a + 16 = 0$.

(b) $a^2 - 8a + 16 = (a-4)^2$. Therefore, there is a double characteristic root, 4.

(c) General solution: $D(k) = (b_{1,0} + b_{1,1}k)\, 4^k$.

(d)

$$\left\{ \begin{array}{c} D(2) = 16 \\ D(3) = 80 \end{array} \right\} \Rightarrow \left\{ \begin{array}{c} (b_{1,0} + b_{1,1}2)\, 4^2 = 16 \\ (b_{1,0} + b_{1,1}3)\, 4^3 = 80 \end{array} \right\}$$
$$\Rightarrow \left\{ \begin{array}{c} 16b_{1,0} + 32b_{1,1} = 16 \\ 64b_{1,0} + 192b_{1,1} = 80 \end{array} \right\} \Rightarrow \left\{ \begin{array}{c} b_{1,0} = \frac{1}{2} \\ b_{1,1} = \frac{1}{4} \end{array} \right\}$$

Therefore $D(k) = (1/2 + (1/4)k)4^k = (2+k)4^{k-1}$.

□

8.3.4 Solution of Nonhomogeneous Finite Order Linear Relations

Our algorithm for nonhomogeneous relations will not be as complete as for the homogeneous case. This is due to the fact that different right-hand sides ($f(k)$'s) call for different rules in obtaining a particular solution.

Algorithm 8.3.16 Algorithm for Solving Nonhomogeneous Finite Order Linear Relations. *To solve the recurrence relation* $S(k) + C_1 S(k-1) + \ldots + C_n S(k-n) = f(k)$

(1) Write the associated homogeneous relation and find its general solution (Steps (a) through (c) of Algorithm 8.3.12). Call this the homogeneous solution, $S^{(h)}(k)$*.*

(2) Start to obtain what is called a particular solution, $S^{(p)}(k)$ *of the recurrence relation by taking an educated guess at the form of a particular solution. For a large class of right-hand sides, this is not really a guess, since the particular solution is often the same type of function as* $f(k)$ *(see Table 8.3.17).*

(3) Substitute your guess from Step 2 into the recurrence relation. If you made a good guess, you should be able to determine the unknown coefficients of your guess. If you made a wrong guess, it should be apparent from the result of this substitution, so go back to Step 2.

(4) The general solution of the recurrence relation is the sum of the homogeneous and particular solutions. If no conditions are given, then you are finished. If n *initial conditions are given, they will translate to* n *linear equations in* n *unknowns and solve the system to get a complete solution.*

Table 8.3.17 Particular solutions for given right-hand sides

Right Hand Side, $f(k)$	Form of Particular Solution, $S^{(p)}(k)$
Constant, q	Constant, d
Linear Function, $q_0 + q_1 k$	Linear Function, $d_0 + d_1 k$
m^{th} degree polynomial, $q_0 + q_1 k + \cdots + q_m k^m$	m^{th} degree polynomial, $d_0 + d_1 k + \cdots + d_m k^m$
exponential function, qa^k	exponential function, da^k

Example 8.3.18 Solution of a Nonhomogeneous First Order Recurrence Relation. Solve $S(k) + 5S(k-1) = 9$, with $S(0) = 6$.

(a) The associated homogeneous relation,$S(k) + 5S(k-1) = 0$ has the characteristic equation $a + 5 = 0$; therefore, $a = -5$. The homogeneous solution is $S^{(h)}(k) = b(-5)^k$.

(b) Since the right-hand side is a constant, we guess that the particular solution will be a constant, d.

(c) If we substitute $S^{(p)}(k) = d$ into the recurrence relation, we get $d + 5d = 9$, or $6d = 9$. Therefore, $S^{(p)}(k) = 1.5$.

(d) The general solution of the recurrence relation is $S(k) = S^{(h)}(k) + S^{(p)}(k) = b(-5)^k + 1.5$ The initial condition will give us one equation to solve in order to determine b. $S(0) = 6 \Rightarrow b(-5)^0 + 1.5 = 6 \Rightarrow b + 1.5 = 6$ Therefore, $b = 4.5$ and $S(k) = 4.5(-5)^k + 1.5$.

□

Example 8.3.19 Solution of a Nonhomogeneous Second Order Recurrence Relation. Consider $T(k) - 7T(k-1) + 10T(k-2) = 6 + 8k$ with $T(0) = 1$ and $T(1) = 2$.

(a) From Example 8.3.13, we know that $T^{(h)}(k) = b_1 2^k + b_2 5^k$. Caution:Don't apply the initial conditions to $T^{(h)}$ until you add $T^{(p)}$!

(b) Since the right-hand side is a linear polynomial, $T^{(p)}$ is linear; that is,

$T^{(p)}(k) = d_0 + d_1 k$.

(c) Substitution into the recurrence relation yields: $(d_0 + d_1 k) - 7(d_0 + d_1(k-1)) + 10(d_0 + d_1(k-2)) = 6 + 8k \quad \Rightarrow (4d_0 - 13d_1) + (4d_1)k = 6 + 8k$ Two polynomials are equal only if their coefficients are equal. Therefore, $\left\{\begin{array}{c} 4d_0 - 13d_1 = 6 \\ 4d_1 = 8 \end{array}\right\} \Rightarrow \left\{\begin{array}{c} d_0 = 8 \\ d_1 = 2 \end{array}\right\}$

(d) Use the general solution $T(k) = b_1 2^k + b_2 5^k + 8 + 2k$ and the initial conditions to get a final solution: $\left\{\begin{array}{c} T(0) = 1 \\ T(1) = 2 \end{array}\right\} \Rightarrow \left\{\begin{array}{c} b_1 + b_2 + 8 = 1 \\ 2b_1 + 5b_2 + 10 = 2 \end{array}\right\}$

$\Rightarrow \left\{\begin{array}{c} b_1 + b_2 = -7 \\ 2b_1 + 5b_2 = -8 \end{array}\right\}$

$\Rightarrow \left\{\begin{array}{c} b_1 = -9 \\ b_2 = 2 \end{array}\right\}$

Therefore, $T(k) = -9 \cdot 2^k + 2 \cdot 5^k + 8 + 2k$.

□

Note 8.3.20 A quick note on interest rates. When a quantity, such as a savings account balance, is increased by some fixed percent, it is most easily computed with a multiplier. In the case of an 8% increase, the multiplier is 1.08 because any original amount A, has $0.08A$ added to it, so that the new balance is $A + 0.08A = (1 + 0.08)A = 1.08A$.

Another example is that if the interest rate is 3.5%, the multiplier would be 1.035. This presumes that the interest is applied at the end of year for 3.5% annual interest, often called **simple interest**. If the interest is applied monthly, and we assume a simplifed case where each month has the same length, the multiplier after every month would be $\left(1 + \frac{0.035}{12}\right) \approx 1.00292$. After a year passes, this multiplier would be applied 12 times, which is the same as multiplying by $1.00292^{12} \approx 1.03557$. That increase from 1.035 to 1.03557 is the effect of **compound interest**.

Example 8.3.21 A Sort of Annuity. Suppose you open a savings account that pays an annual interest rate of 8%. In addition, suppose you decide to deposit one dollar when you open the account, and you intend to double your deposit each year. Let $B(k)$ be your balance after k years. B can be described by the relation $B(k) = 1.08B(k-1) + 2^k$, with $S(0) = 1$. If, instead of doubling the deposit each year, you deposited a constant amount, q, the 2^k term would be replaced with q. A sequence of regular deposits such as this is called a simple annuity.

Returning to the original situation,

(a) $B^{(h)}(k) = b_1(1.08)^k$

(b) $B^{(p)}(k)$ should be of the form $d2^k$.

(c)

$$\begin{aligned} d2^k = 1.08d2^{k-1} + 2^k &\Rightarrow (2d)2^{k-1} = 1.08d2^{k-1} + 2 \cdot 2^{k-1} \\ &\Rightarrow 2d = 1.08d + 2 \\ &\Rightarrow .92d = 2 \\ &\Rightarrow d = 2.174 \text{ to the nearest thousandth)} \end{aligned}$$

Therefore $B^{(p)}(k) = 2.174 \cdot 2^k$.

(d) $B(0) = 1 \Rightarrow\ b_1 + 2.174 = 1$

$\Rightarrow\ b_1 = -1.174$

Therefore, $B(k) = -1.174 \cdot 1.08^k + 2.174 \cdot 2^k$.

□

Example 8.3.22 Matching Roots. Find the general solution to $S(k) - 3S(k-1) - 4S(k-2) = 4^k$.

(a) The characteristic roots of the associated homogeneous relation are -1 and 4. Therefore, $S^{(h)}(k) = b_1(-1)^k + b_2 4^k$.

(b) A function of the form $d4^k$ will not be a particular solution of the nonhomogeneous relation since it solves the associated homogeneous relation. When the right-hand side involves an exponential function with a base that equals a characteristic root,you should multiply your guess at a particular solution by k. Our guess at $S^{(p)}(k)$ would then be $dk4^k$. See Observation 8.3.23 for a more complete description of this rule.

(c) Substitute $dk4^k$ into the recurrence relation for $S(k)$:

$$dk4^k - 3d(k-1)4^{k-1} - 4d(k-2)4^{k-2} = 4^k$$
$$16dk4^{k-2} - 12d(k-1)4^{k-2} - 4d(k-2)4^{k-2} = 4^k$$

Each term on the left-hand side has a factor of 4^{k-2}

$$16dk - 12d(k-1) - 4d(k-2) = 4^2 20d = 16 \Rightarrow d = 0.8$$

Therefore, $S^{(p)}(k) = 0.8k4^k$.

(d) The general solution to the recurrence relation is

$$S(k) = b_1(-1)^k + b_2 4^k + 0.8k4^k$$

□

Observation 8.3.23 When the base of right-hand side is equal to a characteristic root. If the right-hand side of a nonhomogeneous relation involves an exponential with base a, and a is also a characteristic root of multiplicity p, then multiply your guess at a particular solution as prescribed in Table 8.3.17 by k^p, where k is the index of the sequence.

Example 8.3.24 Examples of matching bases.

(a) If $S(k) - 9S(k-1) + 20S(k-2) = 2 \cdot 5^k$, the characteristic roots are 4 and 5. Since 5 matches the base of the right side, $S^{(p)}(k)$ will take the form $dk5^k$.

(b) If $S(n) - 6S(n-1) + 9S(n-2) = 3^{n+1}$ the only characteristic root is 3, but it is a double root (multiplicity 2). Therefore, the form of the particular solution is $dn^2 3^n$.

(c) If $Q(j) - Q(j-1) - 12Q(j-2) = (-3)^j + 6 \cdot 4^j$, the characteristic roots are -3 and 4. The form of the particular solution will be $d_1 j(-3)^j + d_2 j \cdot 4^j$.

(d) If $S(k) - 9S(k-1) + 8S(k-2) = 9k + 1 = (9k+1)1^k$, the characteristic roots are 1 and 8. If the right-hand side is a polynomial, as it is in this case, then the exponential factor 1^k can be introduced. The particular solution will take the form $k\,(d_0 + d_1 k)$.

□

We conclude this section with a comment on the situation in which the characteristic equation gives rise to complex roots. If we restrict the coefficients of our finite order linear relations to real numbers, or even to integers, we can still encounter characteristic equations whose roots are complex. Here, we will simply take the time to point out that our algorithms are still valid with complex characteristic roots, but the customary method for expressing the solutions of these relations is different. Since an understanding of these representations requires some background in complex numbers, we will simply suggest that an interested reader can refer to a more advanced treatment of recurrence relations (see also difference equations).

8.3.5 Exercises

Exercise Group. Solve the following sets of recurrence relations and initial conditions:

1. $S(k) - 10S(k-1) + 9S(k-2) = 0$, $S(0) = 3$, $S(1) = 11$

2. $S(k) - 9S(k-1) + 18S(k-2) = 0$, $S(0) = 0$, $S(1) = 3$

3. $S(k) - 0.25S(k-1) = 0$, $S(0) = 6$

4. $S(k) - 20S(k-1) + 100S(k-2) = 0$, $S(0) = 2$, $S(1) = 50$

5. $S(k) - 2S(k-1) + S(k-2) = 2$, $S(0) = 25$, $S(1) = 16$

6. $S(k) - S(k-1) - 6S(k-2) = -30$, $S(0) = 7$, $S(1) = 6$

7. $S(k) - 5S(k-1) = 5^k$, $S(0) = 3$

8. $S(k) - 5S(k-1) + 6S(k-2) = 2$, $S(0) = -1$, $S(1) = 0$

9. $S(k) - 4S(k-1) + 4S(k-2) = 3k + 2^k$, $S(0) = 1, S(1) = 1$

10. $S(k) = rS(k-1) + a$, $S(0) = 0$, $r, a \geq 0, r \neq 1$

11. $S(k) - 4S(k-1) - 11S(k-2) + 30S(k-3) = 0$, $S(0) = 0$,$S(1) = -35$, $S(2) = -85$

12. Find a closed form expression for $P(k)$ in Exercise 3 of Section 8.2.

13.

(a) Find a closed form expression for the terms of the Fibonacci sequence (see Example 8.1.8).

(b) The sequence C was defined by C_r = the number of strings of zeros and ones with length r having no consecutive zeros (Example 8.2.2(c)). Its recurrence relation is the same as that of the Fibonacci sequence. Determine a closed form expression for C_r, $r \geq 1$.

14. If $S(n) = \sum_{j=1}^{n} g(j)$,$n \geq 1$, then S can be described with the recurrence relation $S(n) = S(n-1) + g(n)$. For each of the following sequences that are defined using a summation, find a closed form expression:

(a) $S(n) = \sum_{j=1}^{n} j$, $n \geq 1$

(b) $Q(n) = \sum_{j=1}^{n} j^2$, $n \geq 1$

(c) $P(n) = \sum_{j=1}^{n} \left(\frac{1}{2}\right)^j$, $n \geq 0$

(d) $T(n) = \sum_{j=1}^{n} j^3$, $n \geq 1$

15. Let $D(n)$ be the number of ways that the set $\{1, 2, ..., n\}$, $n \geq 1$, can be partitioned into two nonempty subsets.

(a) Find a recurrence relation for D. (Hint: It will be a first-order linear relation.)

(b) Solve the recurrence relation.

16. If you were to deposit a certain amount of money at the end of each year for a number of years, this sequence of payments would be called an annuity (see Example 8.3.21).

(a) Find a closed form expression for the balance or value of an annuity that consists of payments of q dollars at a rate of interest of i. Note that for a normal annuity, the first payment is made after one year.

(b) With an interest rate of 5.5 percent, how much would you need to deposit into an annuity to have a value of one million dollars after 18 years?

(c) The payment of a loan is a form of annuity in which the initial value is some negative amount (the amount of the loan) and the annuity ends when the value is raised to zero. How much could you borrow if you can afford to pay \$5,000 per year for 25 years at 11 percent interest?

17. Suppose that C is a small positive number. Consider the recurrence relation $B(k)-2B(k-1)+\left(1-C^2\right)B(k-2)=C^2$, with initial conditions $B(0)=1$ and $B(1)=1$. If C is small enough, we might consider approximating the relation by replacing $1-C^2$ with 1 and C^2 with 0. Solve the original relation and its approximation. Let B_a a be the solution of the approximation. Compare closed form expressions for $B(k)$ and $B_a(k)$. Their forms are very different because the characteristic roots of the original relation were close together and the approximation resulted in one double characteristic root. If characteristic roots of a relation are relatively far apart, this problem will not occur. For example, compare the general solutions of $S(k)+1.001S(k-1)-2.004002S(k-2)=0.0001$ and $S_a(k)+S_a(k-1)-2S_a(k-2)=0$.

8.4 Some Common Recurrence Relations

In this section we intend to examine a variety of recurrence relations that are not finite-order linear with constant coefficients. For each part of this section, we will consider a concrete example, present a solution, and, if possible, examine a more general form of the original relation.

8.4.1 A First Basic Example

Consider the homogeneous first-order linear relation without constant coefficients, $S(n)-nS(n-1)=0$, $n\geq 1$, with initial condition $S(0)=1$. Upon close examination of this relation, we see that the nth term is n times the $(n-1)^{st}$ term, which is a property of n factorial. $S(n)=n!$ is a solution of this relation, for if $n\geq 1$,

$$S(n)=n!=n\cdot(n-1)!=n\cdot S(n-1)$$

In addition, since $0!=1$, the initial condition is satisfied. It should be pointed out that from a computational point of view, our "solution" really isn't much of an improvement since the exact calculation of $n!$ takes $n-1$ multiplications.

If we examine a similar relation, $G(k) - 2^k G(k-1)$, $k \geq 1$ with $G(0) = 1$, a table of values for G suggests a possible solution:

k	0	1	2	3	4	5
$G(k)$	1	2	2^3	2^6	2^{10}	2^{15}

The exponent of 2 in $G(k)$ is growing according to the relation $E(k) = E(k-1) + k$, with $E(0) = 0$. Thus $E(k) = \frac{k(k+1)}{2}$ and $G(k) = 2^{k(k+1)/2}$. Note that $G(k)$ could also be written as $2^0 2^1 2^2 \cdots 2^k$, for $k \geq 0$, but this is not a closed form expression.

In general, the relation $P(n) = f(n)P(n-1)$ for $n \geq 1$ with $P(0) = f(0)$, where f is a function that is defined for all $n \geq 0$, has the "solution"

$$P(n) = \prod_{k=0}^{n} f(k)$$

This product form of $P(n)$ is not a closed form expression because as n grows, the number of multiplications grow. Thus, it is really not a true solution. Often, as for $G(k)$ above, a closed form expression can be derived from the product form.

8.4.2 An Analysis of the Binary Search Algorithm

8.4.2.1

Suppose you intend to use a binary search algorithm (see Subsection 8.1.3) on lists of zero or more sorted items, and that the items are stored in an array, so that you have easy access to each item. A natural question to ask is "How much time will it take to complete the search?" When a question like this is asked, the time we refer to is often the so-called worst-case time. That is, if we were to search through n items, what is the longest amount of time that we will need to complete the search? In order to make an analysis such as this independent of the computer to be used, time is measured by counting the number of steps that are executed. Each step (or sequence of steps) is assigned an absolute time, or weight; therefore, our answer will not be in seconds, but in absolute time units. If the steps in two different algorithms are assigned weights that are consistent, then analyses of the algorithms can be used to compare their relative efficiencies. There are two major steps that must be executed in a call of the binary search algorithm:

(1) If the lower index is less than or equal to the upper index, then the middle of the list is located and its key is compared to the value that you are searching for.

(2) In the worst case, the algorithm must be executed with a list that is roughly half as large as in the previous execution. If we assume that Step 1 takes one time unit and $T(n)$ is the worst-case time for a list of n items, then

$$T(n) = 1 + T(\lfloor n/2 \rfloor), \quad n > 0 \tag{8.4.1}$$

For simplicity, we will assume that

$$T(0) = 0 \tag{8.4.2}$$

even though the conditions of Step 1 must be evaluated as false if $n = 0$. You might wonder why $n/2$ is truncated in (8.4.1). If n is odd, then

$n = 2k + 1$ for some $k \geq 0$, the middle of the list will be the $(k+1)^{st}$ item, and no matter what half of the list the search is directed to, the reduced list will have $k = \lfloor n/2 \rfloor$ items. On the other hand, if n is even, then $n = 2k$ for $k > 0$. The middle of the list will be the k^{th} item, and the worst case will occur if we are directed to the k items that come after the middle (the $(k+1)^{st}$ through $(2k)^{th}$ items). Again the reduced list has $\lfloor n/2 \rfloor$ items.

Solution to (8.4.1) and (8.4.2). To determine $T(n)$, the easiest case is when n is a power of two. If we compute $T\left(2^m\right)$, $m \geq 0$, by iteration, our results are

$$\begin{aligned}
T(1) &= 1 + T(0) = 1 \\
T(2) &= 1 + T(1) = 2 \\
T(4) &= 1 + T(2) = 3 \\
T(8) &= 1 + T(4) = 4
\end{aligned}$$

The pattern that is established makes it clear that $T\left(2^m\right) = m + 1$. This result would seem to indicate that every time you double the size of your list, the search time increases by only one unit.

A more complete solution can be obtained if we represent n in binary form. For each $n \geq 1$, there exists a non-negative integer r such that

$$2^{r-1} \leq n < 2^r \tag{8.4.3}$$

For example, if $n = 21$, $2^4 \leq 21 < 2^5$; therefore, $r = 5$. If n satisfies (8.4c), its binary representation requires r digits. For example, $21_{\text{ten}} = 10101_{\text{two}}$.

In general, $n = \left(a_1 a_2 \ldots a_r\right)_{\text{two}}$. where $a_1 = 1$. Note that in this form, $\lfloor n/2 \rfloor$ is easy to describe: it is the $r-1$ digit binary number $\left(a_1 a_2 \ldots a_{r-1}\right)_{\text{two}}$

Therefore,

$$\begin{aligned}
T(n) &= T\left(a_1 a_2 \ldots a_r\right) \\
&= 1 + T\left(a_1 a_2 \ldots a_{r-1}\right) \\
&= 1 + \left(1 + T\left(a_1 a_2 \ldots a_{r-2}\right)\right) \\
&= 2 + T\left(a_1 a_2 \ldots a_{r-2}\right) \\
&\vdots \\
&= (r-1) + T\left(a_1\right) \\
&= (r-1) + 1 \quad \text{since } T(1) = 1 \\
&= r
\end{aligned}$$

From the pattern that we've just established, $T(n)$ reduces to r. A formal inductive proof of this statement is possible. However, we expect that most readers would be satisfied with the argument above. Any skeptics are invited to provide the inductive proof.

For those who prefer to see a numeric example, suppose $n = 21$.

$$\begin{aligned}
T(21) &= T(10101) \\
&= 1 + T(1010) \\
&= 1 + (1 + T(101)) \\
&= 1 + (1 + (1 + T(10))) \\
&= 1 + (1 + (1 + (1 + T(1)))) \\
&= 1 + (1 + (1 + (1 + (1 + T(0))))) \\
&= 5
\end{aligned}$$

Our general conclusion is that the solution to (8.4.1) and (8.4.2) is that for $n \geq 1$, $T(n) = r$, where $2^{r-1} \leq n < 2^r$.

A less cumbersome statement of this fact is that $T(n) = \lfloor \log_2 n \rfloor + 1$. For example, $T(21) = \lfloor \log_2 21 \rfloor + 1 = 4 + 1 = 5$.

8.4.2.2 Review of Logarithms

Any discussion of logarithms must start by establishing a base, which can be any positive number other than 1. With the exception of Theorem 5, our base will be 2. We will see that the use of a different base (10 and $e \approx 2.171828$ are the other common ones) only has the effect of multiplying each logarithm by a constant. Therefore, the base that you use really isn't very important. Our choice of base 2 logarithms is convenient for the problems that we are considering.

Definition 8.4.1 Base 2 logarithm. The base 2 logarithm of a positive number represents an exponent and is defined by the following equivalence for any positive real numbers a.

$$\log_2 a = x \quad \Leftrightarrow \quad 2^x = a.$$

$\Diamond$

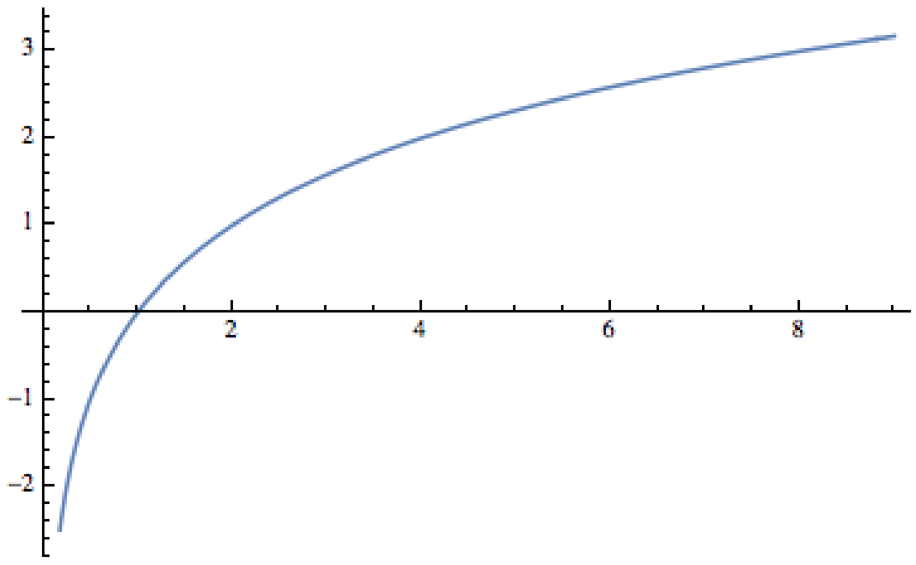

Figure 8.4.2 Plot of the logarithm, bases 2, function

For example, $\log_2 8 = 3$ because $2^3 = 8$ and $\log_2 1.414 \approx 0.5$ because $2^{0.5} \approx 1.414$. A graph of the function $f(x) = \log_2 x$ in Figure 8.4.2 shows that if $a < b$, the $\log_2 a < \log_2 b$; that is, when x increases, $\log_2 x$ also increases. However, if we move x from $2^{10} = 1024$ to $2^{11} = 2048$, $\log_2 x$ only increases from 10 to 11. This slow rate of increase of the logarithm function is an important point to remember. An algorithm acting on n pieces of data that can be executed in $\log_2 n$ time units can handle significantly larger sets of data than an algorithm that can be executed in $n/100$ or $\sqrt{n}$ time units. The graph of $T(n) = \lfloor \log_2 n \rfloor + 1$ would show the same behavior.

A few more properties that we will use in subsequent discussions involving logarithms are summarized in the following theorem.

Theorem 8.4.3 Fundamental Properties of Logarithms. *Let a and b be* *positive real numbers, and r a real number.*

$$\log_2 1 = 0 \tag{8.4.4}$$
$$\log_2 ab = \log_2 a + \log_2 b \tag{8.4.5}$$
$$\log_2 \frac{a}{b} = \log_2 a - \log_2 b \tag{8.4.6}$$
$$\log_2 a^r = r \log_2 a \tag{8.4.7}$$
$$2^{\log_2 a} = a \tag{8.4.8}$$

Definition 8.4.4 Logarithms base b. If $b > 0$, $b \neq 1$, then for $a > 0$,

$$\log_b a = x \Leftrightarrow b^x = a$$

◊

Theorem 8.4.5 How logarithms with different bases are related. *Let $b > 0$, $b \neq 1$. Then for all $a > 0$, $\log_b a = \frac{\log_2 a}{\log_2 b}$. Therefore, if $b > 1$, base b logarithms can be computed from base 2 logarithms by dividing by the positive scaling factor $\log_2 b$. If $b < 1$, this scaling factor is negative.*

Proof. By an analogue of (8.4.8), $a = b^{\log_b a}$. Therefore, if we take the base 2 logarithm of both sides of this equality we get:

$$\log_2 a = \log_2 \left(b^{\log_b a}\right) \Rightarrow \log_2 a = \log_b a \cdot \log_2 b$$

Finally, divide both sides of the last equation by $\log_2 b$. ■

Note 8.4.6 $\log_2 10 \approx 3.32192$ and $\log_2 e \approx 1.4427$.

8.4.2.3

Returning to the binary search algorithm, we can derive the final expression for $T(n)$ using the properties of logarithms, including that the logarithm function is increasing so that inequalities are maintained when taking logarithms of numbers.

$$\begin{aligned} T(n) = r &\Leftrightarrow 2^{r-1} \leq n < 2^r \\ &\Leftrightarrow \log_2 2^{r-1} \leq \log_2 n < \log_2 2^r \\ &\Leftrightarrow r - 1 \leq \log_2 n < r \\ &\Leftrightarrow r - 1 = \lfloor \log_2 n \rfloor \\ &\Leftrightarrow T(n) = r = \lfloor \log_2 n \rfloor + 1 \end{aligned}$$

We can apply several of these properties of logarithms to get an alternate expression for $T(n)$:

$$\begin{aligned} \lfloor \log_2 n \rfloor + 1 &= \lfloor \log_2 n + 1 \rfloor \\ &= \lfloor \log_2 n + \log_2 2 \rfloor \\ &= \lfloor \log_2 2n \rfloor \end{aligned}$$

If the time that was assigned to Step 1 of the binary search algorithm is changed, we wouldn't expect the form of the solution to be very different. If $T(n) = a + T(\lfloor n/2 \rfloor)$ with $T(0) = c$, then $T(n) = c + a \lfloor \log_2 2n \rfloor$.

A further generalization would be to add a coefficient to $T(\lfloor n/2 \rfloor)$: $T(n) = a + bT(\lfloor n/2 \rfloor)$ with $T(0) = c$, where $a, b, c \in \mathbb{R}$, and $b \neq 0$ is not quite as simple to derive. First, if we consider values of n that are powers of 2:

$$\begin{gathered}T(1)=a+bT(0)=a+bc\\ T(2)=a+b(a+bc)=a+ab+cb^2\\ T(4)=a+b\left(a+ab+cb^2\right)=a+ab+ab^2+cb^3\\ \vdots\\ T\left(2^r\right)=a+ab+ab^2+\cdots+ab^r+cb^{r+1}\end{gathered}$$

If n is not a power of 2, by reasoning that is identical to what we used to (8.4.1) and (8.4.2),

$$T(n)=\sum_{k=0}^{r} ab^k+cb^{r+1}$$

where $r=\lfloor\log_2 n\rfloor$.

The first term of this expression is a geometric sum, which can be written in closed form. Let x be that sum:

$$\begin{aligned}x&=a+ab+ab^2+\cdots+ab^r\\ bx&=\quad ab+ab^2+\cdots+ab^r+ab^{r+1}\end{aligned}$$

We've multiplied each term of x by b and aligned the identical terms in x and bx. Now if we subtract the two equations,

$$x-bx=a-ab^{r+1}\Rightarrow x(1-b)=a\left(1-b^{r+1}\right)$$

Therefore, $x=a\frac{b^{r+1}-1}{b-1}$.

A closed form expression for $T(n)$ is

$$T(n)=a\frac{b^{r+1}-1}{b-1}+\ cb^{r+1}\text{ where }r=\lfloor\log_2 n\rfloor$$

8.4.3 Analysis of Bubble Sort and Merge Sort

The efficiency of any search algorithm such as the binary search relies on fact that the search list is sorted according to a key value and that the search is based on the key value. There are several methods for sorting a list. One example is the bubble sort. You might be familiar with this one since it is a popular "first sorting algorithm." A time analysis of the algorithm shows that if $B(n)$ is the worst-case time needed to complete the bubble sort on n items, then $B(n)=(n-1)+B(n-1)$ and $B(1)=0$. The solution of this relation is a quadratic function $B(n)=\frac{1}{2}\left(n^2-n\right)$. The growth rate of a quadratic function such as this one is controlled by its squared term. Any other terms are dwarfed by it as n gets large. For the bubble sort, this means that if we double the size of the list that we are to sort, n changes to $2n$ and so n^2 becomes $4n^2$. Therefore, the time needed to do a bubble sort is quadrupled. One alternative to bubble sort is the merge sort. Here is a simple version of this algorithm for sorting $F=\{r(1),r(2),\ldots,r(n)\}$, $n\geq 1$. If $n=1$, the list is sorted trivially. If $n\geq 2$ then:

(1) Divide F into $F_1=\{r(1),\ldots,r(\lfloor n/2\rfloor)\}$ and $F_2=\{r(\lfloor n/2\rfloor+1),\ldots,r(n)\}$.

(2) Sort F_1 and F_2 using a merge sort.

(3) Merge the sorted lists F_1 and F_2 into one sorted list. If the sort is to be done in descending order of key values, you continue to choose the higher key value from the fronts of F_1 and F_2 and place them in the back of F.

Note that F_1 will always have $\lfloor n/2 \rfloor$ items and F_2 will have $\lceil n/2 \rceil$ items; thus, if n is odd, F_2 gets one more item than F_1. We will assume that the time required to perform Step 1 of the algorithm is insignificant compared to the other steps; therefore, we will assign a time value of zero to this step. Step 3 requires roughly n comparisons and n movements of items from F_1 and F_2 to F; thus, its time is proportional to n. For this reason, we will assume that Step 3 takes n time units. Since Step 2 requires $T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil)$ time units,

$$T(n) = n + T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) \tag{8.4.9}$$

with the initial condition

$$T(1) = 0 \tag{8.4.10}$$

Instead of an exact solution of these equations, we will be content with an estimate for $T(n)$. First, consider the case of $n = 2^r$, $r \geq 1$:

$$\begin{gathered}
T\left(2^1\right) = T(2) = 2 + T(1) + T(1) = 2 = 1 \cdot 2 \\
T\left(2^2\right) = T(4) = 4 + \ T(2) + T(2) = 8 = 2 \cdot 4 \\
T\left(2^3\right) = T(8) = 8 + T(4) + T(4) = 24 = 3 \cdot 8 \\
\vdots \\
T\left(2^r\right) = r 2^r = 2^r \log_2 2^r
\end{gathered}$$

Thus, if n is a power of 2, $T(n) = n \log_2 n$. Now if, for some $r \geq 2$, $2^{r-1} \leq n \leq 2^r$, then $(r-1)2^{r-1} \leq T(n) < r 2^r$. This can be proved by induction on r. As n increases from 2^{r-1} to 2^r, $T(n)$ increases from $(r-1)2^{r-1}$to $r2^r$ and is slightly larger than $\lfloor n \log_2 n \rfloor$. The discrepancy is small enough so that $T_e(n) = \lfloor n \log_2 n \rfloor$ can be considered a solution of (8.4.9) and (8.4.10) for the purposes of comparing the merge sort with other algorithms. Table 8.4.7 compares $B(n)$ with $T_e(n)$ for selected values of n.

Table 8.4.7 Comparison of Times for Bubble Sort and Merge Sort

n	$B(n)$	$T_e(n)$
10	45	34
50	1225	283
100	4950	665
500	124750	4483
1000	499500	9966

8.4.4 Derangements

A derangement is a permutation on a set that has no "fixed points". Here is a formal definition:

Definition 8.4.8 Derangement. A derangement of a nonempty set A is a permutation of A (i.e., a bijection from A into A) such that $f(a) \neq a$ for all $a \in A$. ♢

If $A = \{1, 2, ..., n\}$, an interesting question might be "How many derangements are there of A?" We know that our answer is bounded above by $n!$. We can also expect our answer to be quite a bit smaller than $n!$ since n is the image of itself for $(n-1)!$ of the permutations of A.

Let $D(n)$ be the number of derangements of $\{1, 2, ..., n\}$. Our answer will come from discovering a recurrence relation on D. Suppose that $n \geq 3$. If we are to construct a derangement of $\{1, 2, \ldots, n\}$, f, then $f(n) = k \neq n$. Thus, the image of n can be selected in $n-1$ different ways. No matter which of the $n-1$ choices we make, we can complete the definition of f in one of two ways. First, we can decide to make $f(k)=n$, leaving $D(n-2)$ ways of completing the definition of f, since f will be a derangement of $\{1,2,\ldots,n\}-\{n,k\}$. Second, if we decide to select $f(k)\neq n$, each of the $D(n-1)$ derangements of $\{1,2,\ldots,n-1\}$ can be used to define f. If g is a derangement of $\{1,2,\ldots,n-1\}$ such that $g(p)=k$, then define f by

$$f(j)=\begin{cases} n & \text{if } j=p \\ k & \text{if } j=n \\ g(j) & \text{otherwise} \end{cases}$$

Note that with our second construction of f, $f(f(n))=f(k)\neq n$, while in the first construction, $f(f(n))=f(k)=n$. Therefore, no derangement of $\{1,2,...,n\}$ with $f(n)=k$ can be constructed by both methods.

To recap our result, we see that f is determined by first choosing one of $n-1$ images of n and then constructing the remainder of f in one of $D(n-2)+D(n-1)$ ways. Therefore,

$$D(n)=(n-1)(D(n-2)+D(n-1)) \tag{8.4.11}$$

This homogeneous second-order linear relation with variable coefficients, together with the initial conditions $D(1)=0$ and $D(2)=1$, completely defines D. Instead of deriving a solution of this relation by analytical methods, we will give an empirical derivation of an approximation of $D(n)$. Since the derangements of $\{1,2...,n\}$ are drawn from a pool of $n!$ permutations, we will see what percentage of these permutations are derangements by listing the values of $n!$, $D(n)$, and $\frac{D(n)}{n!}$. The results we observe will indicate that as n grows, $\frac{D(n)}{n!}$ hardly changes at all. If this quotient is computed to eight decimal places, for $n\geq 12$, $D(n)/n! = 0.36787944$. The reciprocal of this number, which $D(n)/n!$ seems to be tending toward, is, to eight places, 2.7182818. This number appears in so many places in mathematics that it has its own name, e. An approximate solution of our recurrence relation on D is then $D(n)\approx\frac{n!}{e}$.

```
def D(n):
    if n<=2:
        return n-1
    else:
        return (n-1)*(D(n-2)+D(n-1))

list(map(lambda
    k:[k,D(k),(D(k)/factorial(k)).n(digits=8)],range(1,16)))
```

```
[[1, 0, 0.00000000],
 [2, 1, 0.50000000],
 [3, 2, 0.33333333],
 [4, 9, 0.37500000],
 [5, 44, 0.36666667],
 [6, 265, 0.36805556],
 [7, 1854, 0.36785714],
 [8, 14833, 0.36788194],
 [9, 133496, 0.36787919],
 [10, 1334961, 0.36787946],
 [11, 14684570, 0.36787944],
 [12, 176214841, 0.36787944],
 [13, 2290792932, 0.36787944],
 [14, 32071101049, 0.36787944],
 [15, 481066515734, 0.36787944]]
```

8.4.5 Exercises

1. Solve the following recurrence relations. Indicate whether your solution is an improvement over iteration.

(a) $nS(n) - S(n-1) = 0$, $S(0) = 1$.

(b) $T(k) + 3kT(k-1) = 0$, $T(0) = 1$.

(c) $U(k) - \frac{k-1}{k}U(k-1) = 0$, $k \geq 2$, $U(1) = 1$.

2. Prove that if $n \geq 0$, $\lfloor n/2 \rfloor + \lceil n/2 \rceil = n$. (Hint: Consider the cases of n odd and n even separately.)

3. Solve as completely as possible:

(a) $T(n) = 3 + T(\lfloor n/2 \rfloor)$, $T(0) = 0$.

(b) $T(n) = 1 + \frac{1}{2}T(\lfloor n/2 \rfloor)$, $T(0) = 2$.

(c) $V(n) = 1 + V\lfloor n/8 \rfloor)$, $V(0) = 0$. (Hint: Write n in octal form.)

4. Prove by induction that if $T(n) = 1 + T(\lfloor n/2 \rfloor)$, $T(0) = 0$, and $2^{r-1} \leq n < 2^r$, $r \geq 1$, then $T(n) = r$.

Hint. Prove by induction on r.

5. Use the substitution $S(n) = T(n+1)/T(n)$ to solve $T(n)T(n-2) = T(n-1)^2$ for $n \geq 2$, with $T(0) = 1$, $T(1) = 6$, and $T(n) \geq 0$.

6. Use the substitution $G(n) = T(n)^2$ to solve $T(n)^2 - T(n-1)^2 = 1$ for $n \geq 1$, with $T(0) = 10$.

7. Solve as completely as possible:

(a) $Q(n) = 1 + Q\left(\lfloor \sqrt{n} \rfloor\right)$, $n \geq 2$, $Q(1) = 0$.

(b) $R(n) = n + R(\lfloor n/2 \rfloor)$, $n \geq 1$, $R(0) = 0$.

8. Suppose Step 1 of the merge sort algorithm did take a significant amount of time. Assume it takes 0.1 time unit, independent of the value of n.

(a) Write out a new recurrence relation for $T(n)$ that takes this factor into account.

(b) Solve for $T\left(2^r\right)$, $r \geq 0$.

(c) Assuming the solution for powers of 2 is a good estimate for all n, compare your result to the solution in the text. As gets large, is there really much difference?

8.5 Generating Functions

This section contains an introduction to the topic of generating functions and how they are used to solve recurrence relations, among other problems. Methods that employ generating functions are based on the concept that you can take a problem involving sequences and translate it into a problem involving generating functions. Once you've solved the new problem, a translation back to sequences gives you a solution of the original problem.

This section covers:

(1) The definition of a generating function.

(2) Solution of a recurrence relation using generating functions to identify the skills needed to use generating functions.

(3) An introduction and/or review of the skills identified in point 2.

(4) Some applications of generating functions.

8.5.1 Definition

Definition 8.5.1 Generating Function of a Sequence. The generating function of a sequence S with terms $S_0, S_1, S_2, \ldots$, is the infinite sum

$$G(S; z) = \sum_{n=0}^{\infty} S_n z^n = S_0 + S_1 z + S_2 z^2 + S_3 z^3 + \cdots$$

The domain and codomain of generating functions will not be of any concern to us since we will only be performing algebraic operations on them. ♢

Example 8.5.2 First Examples.

(a) If $S_n = 3^n, n \geq 0$, then

$$\begin{aligned} G(S; z) &= 1 + 3z + 9z^2 + 27z^3 + \cdots \\ &= \sum_{n=0}^{\infty} 3^n z^n \\ &= \sum_{n=0}^{\infty} (3z)^n \end{aligned}$$

We can get a closed form expression for $G(S; z)$ by observing that $G(S; z) - 3zG(S; z) = 1$. Therefore, $G(S; z) = \frac{1}{1-3z}$.

(b) Finite sequences have generating functions. For example, the sequence of binomial coefficients $\binom{n}{0}, \binom{n}{1}, \ldots, \binom{n}{n}$, $n \geq 1$ has generating function

$$\begin{aligned} G(\binom{n}{\cdot}; z) &= \binom{n}{0} + \binom{n}{1} z + \cdots + \binom{n}{n} z^n \\ &= \sum_{k=0}^{\infty} \binom{n}{k} z^k \\ &= (1+z)^n \end{aligned}$$

by application of the binomial formula.

(c) If $Q(n) = n^2$, $G(Q; z) = \sum_{n=0}^{\infty} n^2 z^n = \sum_{k=0}^{\infty} k^2 z^k$. Note that the index that is used in the summation has no significance. Also, note that the lower limit of the summation could start at 1 since $Q(0) = 0$.

□

8.5.2 Solution of a Recurrence Relation Using Generating Functions

We illustrate the use of generating functions by solving $S(n) - 2S(n-1) - 3S(n-2) = 0$, $n \geq 2$, with $S(0) = 3$ and $S(1) = 1$.

(1) Translate the recurrence relation into an equation about generating functions.

Let $V(n) = S(n) - 2S(n-1) - 3S(n-2)$, $n \geq 2$, with $V(0) = 0$ and $V(1) = 0$. Therefore,

$$G(V;z) = 0 + 0z + \sum_{n=2}^{\infty}(S(n) - 2S(n-1) - 3S(n-2))z^n = 0$$

(2) Solve for the generating function of the unknown sequence, $G(S;z) = \sum_{n=0}^{\infty} S_n z^n$.

$$\begin{aligned} 0 &= \sum_{n=2}^{\infty}(S(n) - 2S(n-1) - 3S(n-2))z^n \\ &= \sum_{n=2}^{\infty} S(n)z^n - 2\left(\sum_{n=2}^{\infty} S(n-1)z^n\right) - 3\left(\sum_{n=2}^{\infty} S(n-2)z^n\right) \end{aligned}$$

Close examination of the three sums above shows:

(a)

$$\begin{aligned} \sum_{n=2}^{\infty} S_n z^n &= \sum_{n=0}^{\infty} S_n z^n - S(0) - S(1)z \\ &= G(S;z) - 3 - z \end{aligned}$$

since $S(0) = 3$ and $S(1) = 1$.

(b)

$$\begin{aligned} \sum_{n=2}^{\infty} S(n-1)z^n &= z\left(\sum_{n=2}^{\infty} S(n-1)z^{n-1}\right) \\ &= z\left(\sum_{n=1}^{\infty} S(n)z^n\right) \\ &= z\left(\sum_{n=0}^{\infty} S(n)z^n - S(0)\right) \\ &= z(G(S;z) - 3) \end{aligned}$$

(c)

$$\begin{aligned} \sum_{n=2}^{\infty} S(n-2)z^n &= z^2\left(\sum_{n=2}^{\infty} S(n-2)z^{n-2}\right) \\ &= z^2 G(S;z) \end{aligned}$$

Therefore,

$$\begin{aligned} &(G(S;z) - 3 - z) - 2z(G(S;z) - 3) - 3z^2 G(S;z) = 0 \\ &\Rightarrow G(S;z) - 2zG(S;z) - 3z^2 G(S;z) = 3 - 5z \\ &\Rightarrow G(S;z) = \frac{3 - 5z}{1 - 2z - 3z^2} \end{aligned}$$

(3) Determine the sequence whose generating function is the one we got in Step 2.

For our example, we need to know one general fact about the closed form expression of an exponential sequence (a proof will be given later):

$$T(n) = ba^n, n \geq 0 \Leftrightarrow G(T; z) = \frac{b}{1 - az} \tag{8.5.1}$$

Now, in order to recognize S in our example, we must write our closed form expression for $G(S; z)$ as a sum of terms like $G(T; z)$ above. Note that the denominator of $G(S; z)$ can be factored:

$$G(S; z) = \frac{3 - 5z}{1 - 2z - 3z^2} = \frac{3 - 5z}{(1 - 3z)(1 + z)}$$

If you look at this last expression for $G(S; z)$ closely, you can imagine how it could be the result of addition of two fractions,

$$\frac{3 - 5z}{(1 - 3z)(1 + z)} = \frac{A}{1 - 3z} + \frac{B}{1 + z} \tag{8.5.2}$$

where A and B are two real numbers that must be determined. Starting on the right of (8.5.2), it should be clear that the sum, for any A and B, would look like the left-hand side. The process of finding values of A and B that make (8.5.2) true is called the **partial fractions decomposition** of the left-hand side:

$$\begin{aligned}\frac{A}{1 - 3z} + \frac{B}{1 + z} &= \frac{A(1 + z)}{(1 - 3z)(1 + z)} + \frac{B(1 - 3z)}{(1 - 3z)(1 + z)} \\ &= \frac{(A + B) + (A - 3B)z}{(1 - 3z)(1 + z)}\end{aligned}$$

Therefore,

$$\left\{ \begin{array}{c} A + B = 3 \\ A - 3B = -5 \end{array} \right\} \Rightarrow \left\{ \begin{array}{c} A = 1 \\ B = 2 \end{array} \right\}$$

and

$$G(S; z) = \frac{1}{1 - 3z} + \frac{2}{1 + z}$$

We can apply (8.5.1) to each term of $G(S; z)$:

- $\frac{1}{1-3z}$ is the generating function for $S_1(n) = 1 \cdot 3^n = 3^n$
- $\frac{2}{1+z}$ is the generating function for $S_2(n) = 2(-1)^n$.

Therefore, $S(n) = 3^n + 2(-1)^n$.

From this example, we see that there are several skills that must be mastered in order to work with generating functions. You must be able to:

(a) Manipulate summation expressions and their indices (in Step 2).

(b) Solve algebraic equations and manipulate algebraic expressions, including partial function decompositions (Steps 2 and 3).

(c) Identify sequences with their generating functions (Steps 1 and 3).

We will concentrate on the last skill first, a proficiency in the other skills is a product of doing as many exercises and reading as many examples as possible.

First, we will identify the operations on sequences and on generating functions.

8.5.3 Operations on Sequences

Definition 8.5.3 Operations on Sequences. Let S and T be sequences of numbers and let c be a real number. Define the sum $S+T$, the scalar product cS, the product ST, the convolution $S*T$, the pop operation $S\uparrow$ (read "S pop"), and the push operation $S\downarrow$ (read "S push") term-wise for $k \geq 0$ by

$$(S+T)(k) = S(k) + T(k) \tag{8.5.3}$$

$$(cS)(k) = cS(k) \tag{8.5.4}$$

$$(S \cdot T)(k) = S(k)T(k) \tag{8.5.5}$$

$$(S*T)(k) = \sum_{j=0}^{k} S(j)T(k-j) \tag{8.5.6}$$

$$(S\uparrow)(k) = S(k+1) \tag{8.5.7}$$

$$(S\downarrow)(k) = \begin{cases} 0 & \text{if } k=0 \\ S(k-1) & \text{if } k>0 \end{cases} \tag{8.5.8}$$

◊

If one imagines a sequence to be a matrix with one row and an infinite number of columns, $S+T$ and cS are exactly as in matrix addition and scalar multiplication. There is no obvious similarity between the other operations and matrix operations.

The pop and push operations can be understood by imagining a sequence to be an infinite stack of numbers with $S(0)$ at the top, $S(1)$ next, etc., as in Figure 8.5.4a. The sequence $S\uparrow$ is obtained by "popping" S(0) from the stack, leaving a stack as in Figure 8.5.4b, with S(1) at the top, S(2) next, etc. The sequence $S\downarrow$ is obtained by placing a zero at the top of the stack, resulting in a stack as in Figure 8.5.4c. Keep these figures in mind when we discuss the pop and push operations.

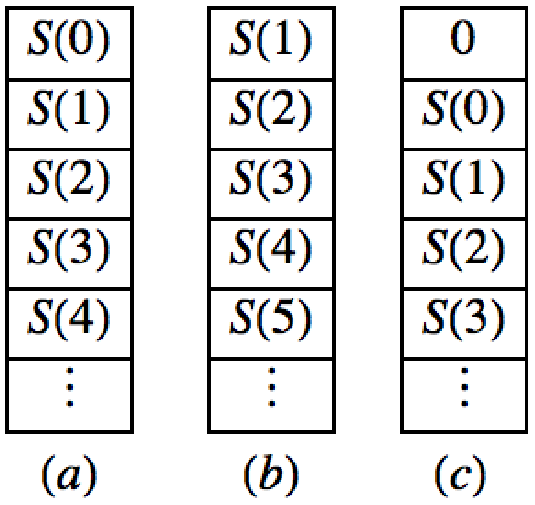

Figure 8.5.4 Stack interpretation of pop and push operation

Example 8.5.5 Some Sequence Operations. If $S(n) = n$, $T(n) = n^2$, $U(n) = 2^n$, and $R(n) = n2^n$:

(a) $(S + T)(n) = n + n^2$

(b) $(U + R)(n) = 2^n + n2^n = (1 + n)2^n$

(c) $(2U)(n) = 2 \cdot 2^n = 2^{n+1}$

(d) $\left(\frac{1}{2}R\right)(n) = \frac{1}{2}n2^n = n2^{n-1}$

(e) $(S \cdot T)(n) = nn^2 = n^3$

(f) $$\begin{aligned}(S * T)(n) &= \sum_{j=0}^{n} S(j)T(n-j) = \sum_{j=0}^{n} j(n-j)^2\\ &= \sum_{j=0}^{n} \left(jn^2 - 2nj^2 + j^3\right)\\ &= n^2\sum_{j=0}^{n} j - 2n\sum_{j=0}^{n} j^2 + \sum_{j=0}^{n} j^3\\ &= n^2\left(\frac{n(n+1)}{2}\right) - 2n\left(\frac{(2n+1)(n+1)n}{6}\right) + \frac{1}{4}n^2(n+1)^2\\ &= \frac{n^2(n+1)(n-1)}{12}\end{aligned}$$

(g) $$\begin{aligned}(U * U)(n) &= \sum_{j=0}^{n} U(j)U(n-j)\\ &= \sum_{j=0}^{n} 2^j 2^{n-j}\\ &= (n+1)2^n\end{aligned}$$

(h) $(S\uparrow)(n) = n + 1$

(i) $(S\downarrow)(n) = \max(0, n-1)$

(j) $((S\downarrow)\downarrow)(n) = \max(0, n-2)$

(k) $(U\downarrow)(n) = \begin{cases} 2^{n-1} & \text{if } n > 0 \\ 0 & \text{if } n = 0 \end{cases}$

(l) $((U\downarrow)\uparrow)(n) = (U\downarrow)(n+1) = 2^n = U(n)$

(m) $((U\uparrow)\downarrow)(n) = \begin{cases} 0 & \text{if } n = 0 \\ U(n) & \text{if } n > 0 \end{cases}$

□

Note that $(U\downarrow)\uparrow \neq (U\uparrow)\downarrow$.

Definition 8.5.6 Multiple Pop and Push. If S is a sequence of numbers and p a positive integer greater than 1, define

$$S\uparrow p = (S\uparrow(p-1))\uparrow \quad \text{if } p \geq 2 \text{ and } S\uparrow 1 = S\uparrow$$

Similarly, define

$$S\downarrow p = (S\downarrow(p-1))\downarrow \quad \text{if } p \geq 2 \text{ and } S\downarrow 1 = S\downarrow$$

◊

In general, $(S\uparrow p)(k) = S(k+p)$, and

$$(S\downarrow p)(k) = \begin{cases} 0 & \text{if } k < p \\ S(k-p) & \text{if } k \geq p \end{cases}$$

8.5.4 Operations on Generating Functions

Definition 8.5.7 Operations on Generating Functions. If $G(z) = \sum_{k=0}^{\infty} a_k z^k$ and $H(z) = \sum_{k=0}^{\infty} b_k z^k$ are generating functions and c is a real number, then the sum $G+H$, scalar product cG, product GH, and monomial product $z^p G$, $p \geq 1$ are generating functions, where

$$(G+H)(z) = \sum_{k=0}^{\infty} (a_k + b_k)\, z^k \tag{8.5.9}$$

$$(cG)(z) = \sum_{k=0}^{\infty} c a_k z^k \tag{8.5.10}$$

$$(GH)(z) = \sum_{k=0}^{\infty} c z^k \text{ where } c_k = \sum_{j=0}^{k} a_j b_{k-j} \tag{8.5.11}$$

$$(z^p G)(z) = z^p \sum_{k=0}^{\infty} a_k z^k = \sum_{k=0}^{\infty} a_k z^{k+p} = \sum_{n=p}^{\infty} a_{n-p} z^n \tag{8.5.12}$$

The last sum is obtained by substituting $n-p$ for k in the previous sum. ◊

Example 8.5.8 Some operations on generating functions. If $D(z) = \sum_{k=0}^{\infty} k z^k$ and $H(z) = \sum_{k=0}^{\infty} 2^k z^k$ then

$$(D+H)(z) = \sum_{k=0}^{\infty} \left(k + 2^k\right) z^k$$

$$(2H)(z) = \sum_{k=0}^{\infty} 2 \cdot 2^k z^k = \sum_{k=0}^{\infty} 2^{k+1} z^k$$

$$\begin{aligned}(zD)(z) &= z\sum_{k=0}^{\infty} kz^k = \sum_{k=0}^{\infty} kz^{k+1}\\ &= \sum_{k=1}^{\infty}(k-1)z^k = D(z) - \sum_{k=1}^{\infty} z^k\end{aligned}$$

$$(DH)(z) = \sum_{k=0}^{\infty}\left(\sum_{j=0}^{k} j2^{k-j}\right)z^k$$

$$(HH)(z) = \sum_{k=0}^{\infty}\left(\sum_{j=0}^{k} 2^j 2^{k-j}\right)z^k = \sum_{k=0}^{\infty}(k+1)2^k z^k$$

Note: $D(z) = G(S;z)$, and $H(z) = G(U;z)$ from Example 5. □

Now we establish the connection between the operations on sequences and generating functions. Let S and T be sequences and let c be a real number.

$$G(S+T;z) = G(S;z) + G(T;z) \tag{8.5.13}$$
$$G(cS;z) = cG(S;z) \tag{8.5.14}$$
$$G(S*T;z) = G(S;z)G(T;z) \tag{8.5.15}$$
$$G(S\uparrow;z) = (G(S;z) - S(0))/z \tag{8.5.16}$$
$$G(S\downarrow;z) = zG(S;z) \tag{8.5.17}$$

In words, (8.5.13) says that the generating function of the sum of two sequences equals the sum of the generating functions of those sequences. Take the time to write out the other four identities in your own words. From the previous examples, these identities should be fairly obvious, with the possible exception of the last two. We will prove (8.5.16) as part of the next theorem and leave the proof of (8.5.17) to the interested reader. Note that there is no operation on generating functions that is related to sequence multiplication; that is, $G(S\cdot T;z)$ cannot be simplified.

Theorem 8.5.9 Generating functions related to Pop and Push. *If $p > 1$,*

(a) $G(S\uparrow p;z) = \left(G(S;z) - \sum_{k=0}^{p-1} S(k)z^k\right)/z^k$

(b) $G(S\downarrow p;z) = z^pG(S;z)$.

Proof. We prove (a) by induction and leave the proof of (b) to the reader.

Basis:

$$\begin{aligned}G(S\uparrow;z) &= \sum_{k=0}^{\infty} S(k+1)z^k\\ &= \sum_{k=1}^{\infty} S(k)z^{k-1}\\ &= \left(\sum_{k=1}^{\infty} S(k)z^k\right)\Bigg/ z\\ &= \left(S(0) + \sum_{k=1}^{\infty} S(k)z^k - S(0)\right)\Bigg/ z\\ &= (G(S;z) - S(0))/z\end{aligned}$$

Therefore, part (a) is true for $p = 1$.

Induction: Suppose that for some $p \geq 1$, the statement in part (a) is true:

$$\begin{aligned} G(S\uparrow(p+1);z) &= G((S\uparrow p)\uparrow;z) \\ &= (G(S\uparrow p;z)-(S\uparrow p)(0))/z \text{ by the basis} \\ &= \frac{\frac{\left(G(S;z)-\sum_{k=0}^{p-1}S(k)z^k\right)}{z^p}-S(p)}{z} \end{aligned}$$

by the induction hypothesis. Now write $S(p)$ in the last expression above as $(S(p)z^p)/z^p$ so that it fits into the finite summation:

$$\begin{aligned} G(S\uparrow(p+1);z) &= \left(\frac{G(S;z)-\sum_{k=0}^{p}S(k)z^k}{z^p}\right)\Big/ z \\ &= \left(G(S;z)-\sum_{k=0}^{p}S(k)z^k\right)/z^{p+1} \end{aligned}$$

Therefore the statement is true for $p+1$. ■

8.5.5 Closed Form Expressions for Generating Functions

The most basic tool used to express generating functions in closed form is the closed form expression for the geometric series, which is an expression of the form $a+ar+ar^2+\cdots$. It can either be terminated or extended infinitely.

Finite Geometric Series:

$$a+ar+ar^2+\cdots+ar^n = a\left(\frac{1-r^{n+1}}{1-r}\right) \tag{8.5.18}$$

Infinite Geometric Series:

$$a+ar+ar^2+\cdots = \frac{a}{1-r} \tag{8.5.19}$$

Restrictions: a and r represent constants and the right sides of the two equations apply under the following conditions:

(1) r must not equal 1 in the finite case. Note that $a+ar+\cdots ar^n = (n+1)a$ if $r=1$.

(2) In the infinite case, the absolute value of r must be less than 1.

These restrictions don't come into play with generating functions. We could derive (8.5.18) by noting that if $S(n) = a+ar+\cdots+ar^n$, $n>0$, then $S(n) = rS(n-1)+a$ (See Exercise 10 of Section 8.3). An alternative derivation was used in Section 8.4. We will take the same steps to derive (8.5.19). Let $x = a+ar+ar^2+\cdots$. Then

$$rx = ar+ar^2+\cdots = x-a \Rightarrow x-rx = a \Rightarrow x = \frac{a}{1-r}$$

Example 8.5.10 Generating Functions involving Geometric Sums.

(a) If $S(n) = 9\cdot 5^n$, $n\geq 0$, $G(S;z)$ is an infinite geometric series with $a=9$ and $r=5z$.Therefore, $G(S;z) = \frac{9}{1-5z}$.

(b) If $T(n)=4$, $n \geq 0$, then $G(T;z) = 4/(1-z)$.

(c) If $U(n) = 3(-1)^n$, then $G(U;z) = 3/(1+z)$.

(d) Let $C(n) = S(n) + T(n) + U(n) = 9 \cdot 5^n + 4 + 3(-1)^n$. Then

$$\begin{aligned} G(C;z) &= G(S;z) + G(T;z) + G(U;z) \\ &= \frac{9}{1-5z} + \frac{4}{1-z} + \frac{3}{1+z} \\ &= -\frac{14z^2 + 34z - 16}{5z^3 - z^2 - 5z + 1} \end{aligned} .$$

Given a choice between the last form of $G(C;z)$ and the previous sum of three fractions, we would prefer leaving it as a sum of three functions. As we saw in an earlier example, a partial fractions decomposition of a fraction such as the last expression requires some effort to produce.

(e) If $G(Q;z) = 34/(2-3z)$, then Q can be determined by multiplying the numerator and denominator by 1/2 to obtain $\frac{17}{1-\frac{3}{2}z}$. We recognize this fraction as the sum of the infinite geometric series with $a = 17$ and $r = \frac{3}{2}z$. Therefore $Q(n) = 17(3/2)^n$.

(f) If $G(A;z) = (1+z)^3$, then we expand $(1+z)^3$ to $1 + 3z + 3z^2 + z^3$. Therefore $A(0) = 1$, $A(1) = 3$ $A(2) = 3$, $A(3) = 1$, and, since there are no higher-powered terms, $A(n) = 0$, $n \geq 4$. A more concise way of describing A is $A(k) = \binom{3}{k}$, since $\binom{n}{k}$ is interpreted as 0 of $k > n$.

□

Table 8.5.11 lists some closed form expressions for the generating functions of some common sequences.

Table 8.5.11 Closed Form Expressions of some Generating Functions

Sequence	Generating Function
$S(k) = ba^k$	$G(S;z) = \frac{b}{1-az}$
$S(k) = k$	$G(S;z) = \frac{z}{(1-z)^2}$
$S(k) = bka^k$	$G(S;z) = \frac{abz}{(1-az)^2}$
$S(k) = \frac{1}{k!}$	$G(S;z) = e^z$
$S(k) = \begin{cases} \binom{n}{k} & 0 \leq k \leq n \\ 0 & k > n \end{cases}$	$G(S;z) = (1+z)^n$

Example 8.5.12 Another Complete Solution. Solve $S(k) + 3S(k-1) - 4S(k-2) = 0$, $k \geq 2$, with $S(0) = 3$ and $S(1) = -2$. The solution will be derived using the same steps that were used earlier in this section, with one variation.

(1) Translate to an equation about generating functions. First, we change the index of the recurrence relation by substituting $n+2$ for k. The result is $S(n+2) + 3S(n+1) - 4S(n) = 0$, $n \geq 0$. Now, if $V(n) = S(n+2) + 3S(n+1) - 4S(n)$, then V is the zero sequence, which has a zero generating function. Furthermore, $V = S \uparrow 2 + 3(S \uparrow) - 4S$.

Therefore,

$$\begin{aligned}0 &= G(V;z)\\ &= G(S\uparrow 2;z) + 3G(S\uparrow;z) - 4G(S;z)\\ &= \frac{G(S;z)-S(0)-S(1)z}{z^2} + 4\frac{(G(S;z)-S(0))}{z} - 4G(S;z)\end{aligned}$$

(2) We want to now solve the following equation for $G(S;z)$:

$$\frac{G(S;z)-S(0)-S(1)z}{z^2} + 4\frac{(G(S;z)-S(0))}{z} - 4G(S;z) = 0$$

Multiply by z^2 :

$$G(S;z) - 3 + 2z + 3z(G(S;z) - 3) - 4z^2G(S;z) = 0$$

Expand and collect all terms involving $G(S;z)$ on one side of the equation:

$$\begin{aligned}G(S;z) + 3zG(S;z) - 4z^2G(S;z) &= 3 + 7z\\ \left(1+3z-4z^2\right)G(S;z) &= 3+7z\end{aligned}$$

Therefore,

$$G(S;z) = \frac{3+7z}{1+3z-4z^2}$$

(3) Determine S from its generating function. $1+3z-4z^2 = (1+4z)(1-z)$ thus a partial fraction decomposition of $G(S;z)$ would be:

$$\frac{A}{1+4z} + \frac{B}{1-z} = \frac{Az - A - 4Bz - B}{(z-1)(4z+1)} = \frac{(A+B)+(4B-A)z}{(z-1)(4z+1)}$$

Therefore, $A+B=3$ and $4B-A=7$. The solution of this set of equations is $A=1$ and $B=2$. $G(S;z) = \frac{1}{1+4z} + \frac{2}{1-z}$.

$\frac{1}{1+4z}$ is the generating function of $S_1(n) = (-4)^n$, and
$\frac{2}{1-z}$ is the generating function of $S_2(n) = 2(1)^n = 2$

In conclusion, since $G(S;z) = G(S_1;z) + G(S_2;z)$, $S(n) = 2 + (-4)^n$.

□

Example 8.5.13 An Application to Counting. Let $A = \{a,b,c,d,e\}$ and let A^* be the set of all strings of length zero or more that can be made using each of the elements of A zero or more times. By the generalized rule of products, there are 5^n such strings that have length n, $n \geq 0$, Suppose that X_n is the set of strings of length n with the property that all of the a's and b's precede all of the c's, d's, and e's. Thus $aaabde \in X_6$, but $abcabc \notin X_6$. Let $R(n) = |X_n|$. A closed form expression for R can be obtained by recognizing R as the convolution of two sequences. To illustrate our point, we will consider the calculation of $R(6)$.

Note that if a string belongs to X_6, it starts with k characters from $\{a,b\}$ and is followed by $6-k$ characters from $\{c,d,e\}$. Let $S(k)$ be the number of strings of a's and b's with length k and let $T(k)$ be the number of strings of c's, d's, and e's with length k. By the generalized rule of products, $S(k) = 2^k$ and $T(k) = 3^k$. Among the strings in X_6 are the ones that start with two a's and b's and end with c's, d's, and e's. There are $S(2)T(4)$ such strings. By the law of addition,

$$|X_6| = R(6) = S(0)T(6) + S(1)T(5) + \cdots + S(5)T(1) + S(6)T(0)$$

Note that the sixth term of R is the sixth term of the convolution of S with T, $S * T$. Think about the general situation for a while and it should be clear that $R = S * T$. Now, our course of action will be to:

(a) Determine the generating functions of S and T,

(b) Multiply $G(S; z)$ and $G(T; z)$ to obtain $G(S * T; z) = G(R; z)$, and

(c) Determine R on the basis of $G(R; z)$.

(a) $G(S; z) = \sum_{k=0}^{\infty} 2^k z^k = \frac{1}{1-2z}$, and $G(T; z) = \sum_{k=0}^{\infty} 3^k z^k = \frac{1}{1-3z}$

(b) $G(R; z) = G(S; z)G(T; z) = \dfrac{1}{(1-2z)(1-3z)}$

(c) To recognize R from $G(R; z)$, we must do a partial fractions decomposition:

$$\frac{1}{(1-2z)(1-3z)} = \frac{A}{1-2z} + \frac{B}{1-3z} = \frac{-3Az + A - 2Bz + B}{(2z-1)(3z-1)} = \frac{(A+B) + (-3A-2B)z}{(2z-1)(3z-1)}$$

Therefore, $A + B = 1$ and $-3A - 2B = 0$. The solution of this pair of equations is $A = -2$ and $B = 3$. Since $G(R; z) = \frac{-2}{1-2z} + \frac{3}{1-3z}$, which is the sum of the generating functions of $-2(2)^k$ and $3(3)^k$, $R(k) = -2(2)^k + 3(3)^k = 3^{k+1} - 2^{k+1}$

For example, $R(6) = 3^7 - 2^7 = 2187 - 128 = 2059$. Naturally, this equals the sum that we get from $(S * T)(6)$. To put this number in perspective, the total number of strings of length 6 with no restrictions is $5^6 = 15625$, and $\frac{2059}{15625} \approx 0.131776$. Therefore approximately 13 percent of the strings of length 6 satisfy the conditions of the problem.

□

8.5.6 Extra for Experts

The remainder of this section is intended for readers who have had, or who intend to take, a course in combinatorics. We do not advise that it be included in a typical course. The method that was used in the previous example is a very powerful one and can be used to solve many problems in combinatorics. We close this section with a general description of the problems that can be solved in this way, followed by some examples.

Consider the situation in which P_1, P_2, …, P_m are m actions that must be taken, each of which results in a well-defined outcome. For each $k = 1, 2, ..., m$ define X_k to be the set of possible outcomes of P_k . We will assume that each outcome can be quantified in some way and that the quantification of the elements of X_k is defined by the function $Q_k : X_k \to \{0, 1, 2, ...\}$. Thus, each outcome has a non-negative integer associated with it. Finally, define a frequency function $F_k : \{0, 1, 2, ...\} \to \{0, 1, 2, ...\}$ such that $F_k(n)$ is the number of elements of X_k that have a quantification of n.

Now, based on these assumptions, we can define the problems that can be solved. If a process P is defined as a sequence of actions $P_1, P_2, \ldots, P_m$ as above, and if the outcome of P, which would be an element of $X_1 \times X_2 \times \cdots \times X_m$, is quantified by

$$Q\left(a_1, a_2, \ldots, a_m\right)=\sum_{k=1}^m Q_k\left(a_k\right)$$

then the frequency function, F, for P is the convolution of the frequency functions forP_1, P_2, …, P_m, which has a generating function equal to the product of the generating functions of the frequency functions F_1, F_2, …, F_m. That is,

$$G(F;z)=G\left(F_1;z\right)G\left(F_2;z\right)\cdots\left(F_m;z\right)$$

Example 8.5.14 Rolling Two Dice. Suppose that you roll a die two times and add up the numbers on the top face for each roll. Since the faces on the die represent the integers 1 through 6, the sum must be between 2 and 12. How many ways can any one of these sums be obtained? Obviously, 2 can be obtained only one way, with two 1's. There are two sequences that yield a sum of 3: 1-2 and 2-1. To obtain all of the frequencies with which the numbers 2 through 12 can be obtained, we set up the situation as follows. For $j=1,2$; P_j is the rolling of the die for the j^{th} time. $X_j=\{1,2,...,6\}$ and $Q_j: X_j\to\{0,1,2,3,\ldots\}$ is defined by $Q_j(x)=x$. Since each number appears on a die exactly once, the frequency function is $F_j(k)=1$ if $1\leq k\leq 6$, and $F_j(k)=0$ otherwise. The process of rolling the die two times is quantified by adding up the $Q_j{}'s$; that is, $Q\left(a_1,a_2\right)=Q_1\left(a_1\right)+Q_2\left(a_2\right)$. The generating function for the frequency function of rolling the die two times is then

$$\begin{aligned}G(F;z)&=G\left(F_1;z\right)G\left(F_2;z\right)\\&=\left(z^6+z^5+z^4+z^3+z^2+z\right)^2\\&=z^{12}+2z^{11}+3z^{10}+4z^9+5z^8+6z^7+5z^6+4z^5+3z^4+2z^3+z^2\end{aligned}$$

Now, to get $F(k)$, just read the coefficient of z^k. For example, the coefficient of z^5 is 4, so there are four ways to roll a total of 5.

To apply this method, the crucial step is to decompose a large process in the proper way so that it fits into the general situation that we've described. □

Example 8.5.15 Distribution of a Committee. Suppose that an organization is divided into three geographic sections, A, B, and C. Suppose that an executive committee of 11 members must be selected so that no more than 5 members from any one section are on the committee and that Sections A, B, and C must have minimums of 3, 2, and 2 members, respectively, on the committee. Looking only at the number of members from each section on the committee, how many ways can the committee be made up? One example of a valid committee would be 4 A's, 4 B's, and 3 C's.

Let P_A be the action of deciding how many members (not who) from Section A will serve on the committee. $X_A=\{3,4,5\}$ and $Q_A(k)=k$. The frequency function, F_A , is defined by $F_A(k)=1$ if $k\in X_k$, with $F_A(k)=0$ otherwise. $G\left(F_A;z\right)$ is then $z^3+z^4+z^5$. Similarly, $G\left(F_B;z\right)=z^2+z^3+z^4+z^5=G\left(F_C;z\right)$. Since the committee must have 11 members, our answer will be the coefficient of z^{11} in $G\left(F_A;z\right)G\left(F_B;z\right)G\left(F_C;z\right)$, which is 10.

```
%display latex
var('z')
expand((z^3+ z^4+z^5)*(z^2+ z^3+ z ^4 + z^5)^2)
```

```
z^15 + 3*z^14 + 6*z^13 + 9*z^12 + 10*z^11 + 9*z^10 + 6*z^9 +
    3*z^8 + z^7
```

□

8.5.7 Exercises

1. What sequences have the following generating functions?

(a) 1

(b) $\dfrac{10}{2-z}$

(c) $1+z$

(d) $\dfrac{3}{1+2z}+\dfrac{3}{1-3z}$

2. What sequences have the following generating functions?

(a) $\dfrac{1}{1+z}$

(b) $\dfrac{1}{4-3z}$

(c) $\dfrac{2}{1-z}+\dfrac{1}{1+z}$

(d) $\dfrac{z+2}{z+3}$

3. Find closed form expressions for the generating functions of the following sequences:

(a) $V(n) = 9^n$

(b) P, where $P(k) - 6P(k-1) + 5P(k-2) = 0$ for $k \geq 2$, with $P(0) = 2$ and $P(1) = 2$.

(c) The Fibonacci sequence: $F(k+2) = F(k+1) + F(k)$, $k \geq 0$, with $F(0) = F(1) = 1$.

4. Find closed form expressions for the generating functions of the following sequences:

(a) $W(n) = \binom{5}{n}2^n$ for $0 \leq n \leq 5$ and $W(n) = 0$ for $n > 5$.

(b) Q, where $Q(k) + Q(k-1) - 42Q(k-2) = 0$ for $k \geq 2$, with $Q(0) = 2$ and $Q(1) = 2$.

(c) G, where $G(k+3) = G(k+2) + G(k+1) + G(k)$ for $k \geq 0$, with $G(0) = G(1) = G(2) = 1$.

5. For each of the following expressions, find the partial fraction decomposition and identify the sequence having the expression as a generating function.

(a) $\dfrac{5+2z}{1-4z^2}$

(b) $\dfrac{32-22z}{2-3z+z^2}$

(c) $\dfrac{6-29z}{1-11z+30z^2}$

6. Find the partial fraction decompositions and identify the sequence having the following expressions:

(a) $\dfrac{1}{1-9z^2}$

(b) $\dfrac{1+3z}{16-8z+z^2}$

(c) $\dfrac{2z}{1-6z-7z^2}$

7. Given that $S(k) = k$ and $T(k) = 10k$, what is the k^{th} term of the generating function of each of the following sequences:

(a) $S + T$

(b) $S\uparrow * T$

(c) $S * T$

(d) $S\uparrow * S\uparrow$

8. Given that $P(k) = \binom{10}{k}$ and $Q(k) = k!$, what is the k^{th} term of the generating function of each of the following sequences:

(a) $P * P$

(b) $P + P\uparrow$

(c) $P * Q$

(d) $Q * Q$

9. A game is played by rolling a die five times. For the k^{th} roll, one point is added to your score if you roll a number higher than k. Otherwise, your score is zero for that roll. For example, the sequence of rolls $2, 3, 4, 1, 2$ gives you a total score of three; while a sequence of 1,2,3,4,5 gives you a score of zero. Of the $6^5 = 7776$ possible sequences of rolls, how many give you a score of zero?, of one? … of five?

10. Suppose that you roll a die ten times in a row and record the square of each number that you roll. How many ways could the sum of the squares of your rolls equal 40? What is the most common outcome?