Chapter 2

Solving Recurrences

This chapter is predominantly concerned with solving a linear recurrence with constant coefficients.

The first approach is completely general:

> one derives a **generating function** for the sequence specified by the recurrence, and then one analyzes that generating function so as to **derive a closed form for the values in the sequence.**

The second approach is restricted to linear recurrence relations with constant coefficients:

> using prior memorization or a table of standard patterns, **one sees how a given linear recurrence fits a standard pattern and adapts the solution.**

How to solve simultaneous recurrences is described in §2.5. Special properties of the Fibonacci numbers are featured in §2.6. The focus of §2.7 and §2.8 is on techniques for transforming a more complicated type of recurrence into a linear recurrence with constant coefficients, thus preconditioning it for solution by the well-established methods of the earlier sections.

2.1 TYPES OF RECURRENCES

REVIEW FROM §1.2:

- A ***recurrence*** prescribes a set of ***initial values***

$$x_0 = b_0 \quad x_1 = b_1 \quad \dots \quad x_k = b_k$$

and a ***recursion*** formula

$$x_n \;=\; \phi(x_{n-1},\; x_{n-2},\; \dots,\; x_0) \qquad \text{for } n > k$$

from which one may calculate the value of x_n, for any $n > k$, from the values of earlier entries.

DEF: A recursion formula of the form

$$x_n \;=\; a_{n-1}x_{n-1} + a_{n-2}x_{n-2} + \cdots + a_0x_0 + \alpha(n)$$

in which each term is linear is a ***linear recursion.*** Each coefficient a_j may be either a ***constant coefficent***, the same for all n, or a function of n, that is, a ***variable coefficient.***

- It is a ***recursion of degree*** d if the number of coefficients a_j that are non-zero is bounded, and if the smallest subscript among the non-zero coefficients is $n - d$.
- The function $\alpha(n)$ is the ***particularity function.***
- It is a ***homogeneous recursion*** if $\alpha(n) = 0$.

Some Linear Recursions

Linear recursions are usually easier to solve than non-linear recursions.

Example 2.1.1: The *Tower of Hanoi recursion* (introduced in §1.2 and solved in §2.2)

$$h_n = 2h_{n-1} + 1 \tag{2.1.1}$$

is a non-homogeneous, linear recursion of degree 1, with a constant coefficient.

Example 2.1.2: The *Fibonacci recursion* (introduced in §0.2 and §1.2 and solved in §2.5)

$$f_n = f_{n-1} + f_{n-2} \tag{2.1.2}$$

is a homogenous linear recursion of degree 2, with constant coefficients.

Recurrences without Fixed Degree

A recurrence of fixed degree d for a sequence $\langle x_n \rangle$ prescribes x_n as a combination of the recent past entries

$$x_{n-1} \quad x_{n-2} \quad \cdots \quad x_{n-d}$$

In the most important kind of recurrence without fixed degree, the value of x_n is a combination of entries whose indices are a fraction of n. This is called a *divide-and-conquer* recurrence.

Example 2.1.3: The *merge-sort recurrence* (explained and solved in §2.8)

$$\begin{aligned} m_1 &= 1; \\ m_n &= 2m_{\lceil \frac{n}{2} \rceil} + n \end{aligned} \tag{2.1.3}$$

is non-homogeneous linear recurrence, without fixed degree, with a constant coefficient. Its recursion formula expresses that the problem of sorting a list of length n is reduced to merging two lists of size $\frac{n}{2}$.

Variable Coefficients

The three recursions (2.1.1), (2.1.2), and (2.1.3) all have constant coefficients. One of the most important linear recursions with variable coefficients arises in the study of *permutations*.

REVIEW FROM §0.5:

- A ***permutation*** on a set S is a one-to-one, onto function from S to itself.
- **Theorem 0.5.3.** Every permutation is a composition of disjoint cyclic permutations.

DEF: A ***derangement*** is a permutation π with no fixed points. That is, there is no object x such that $\pi(x) = x$.

Example 2.1.4: Figure 2.1.1 illustrates a derangement.

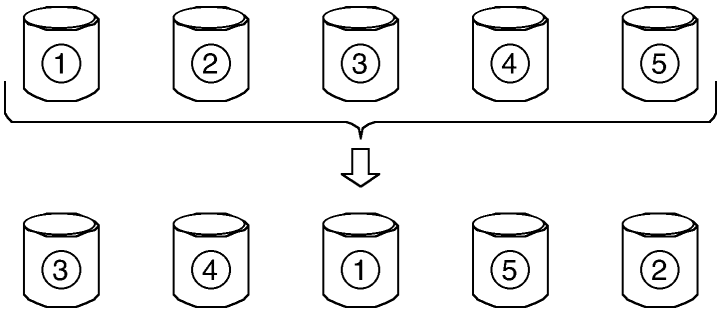


Fig 2.1.1 The derangement $(1\ 3)\,(2\ 5\ 4)$**.**

DEF: The ***derangement number*** D_n is the number of derangements of the integer interval $[1:n]$.

The derangements of the smallest integer intervals $[1:n]$ are given in Table 2.1.1. We observe that a derangement has no 1-cycle in its disjoint cycle form.

Table 2.1.1 Derangements of Small Intervals $[1:n]$**.**

n				D_n
1				0
2	$(1\ 2)$			1
3	$(1\ 2\ 3)$	$(1\ 3\ 2)$		2
4	$(1\ 2)(3\ 4)$	$(1\ 3)(2\ 4)$	$(1\ 4)(2\ 3)$	9
	$(1\ 2\ 3\ 4)$	$(1\ 2\ 4\ 3)$	$(1\ 3\ 2\ 4)$	
	$(1\ 3\ 4\ 2)$	$(1\ 4\ 2\ 3)$	$(1\ 4\ 3\ 2)$	

Prop 2.1.1. *The derangement numbers D_n satisfy the following recursion formula.*

$$D_n \;=\; (n-1)D_{n-1} \,+\, (n-1)D_{n-2} \qquad (2.1.4)$$

Proof: Every derangement of $[1:n]$ such that n does *not* lie in a 2-cycle can be formed by inserting the number n, immediately after one of the $n-1$ numbers in some cycle of some derangement of $[1:n-1]$.

Every derangement of $[1:n]$ in which n *does* lie in a 2-cycle can be formed from some derangement π of $[1:n-2]$ either by adding the 2-cycle $(n-1 \;\; n)$, or by replacing one of the $n-2$ numbers j in some cycle of π by the number $n-1$ and then adding the 2-cycle $(j \;\; n)$. $\diamondsuit$

Example 2.1.5: The *derangement recurrence* (considered in more detail in §5.4)

$$\begin{aligned} D_0 \;&=\; 1, \quad D_1 \;=\; 0; \\ D_n \;&=\; (n-1)D_{n-1} \,+\, (n-1)D_{n-2} \end{aligned}$$

is homogenous, linear, of degree 2, whose coefficients are variable. The sequence it specifies is convex, since

$$\begin{aligned} \frac{D_{n+1} \,+\, D_{n-1}}{2} \;&=\; \frac{(nD_n + nD_{n-1}) \,+\, D_{n-1}}{2} \\ &\geq\; \frac{n}{2}\, D_n \\ &\geq\; D_n \quad \text{for } n \geq 2 \end{aligned}$$

Even without solving the derangement recurrence, it is possible to prove inductively that most permutations have a fixed point, that is, that the ratio $\frac{D_n}{n!}$ of derangements to permutations is less than half.

BASIS: This is clearly true for $n = 1$ and $n = 2$.

IND STEP: For $n \geq 3$, we have

$$\begin{aligned} D_n &= (n-1)D_{n-1} + (n-1)D_{n-2} \\ &< (n-1)\frac{(n-1)!}{2} + (n-1)\frac{(n-2)!}{2} \qquad \text{(ind hyp)} \\ &= n\frac{(n-1)!}{2} - \frac{(n-1)!}{2} + (n-1)\frac{(n-2)!}{2} \\ &= \frac{n!}{2} \end{aligned}$$

Some Non-linear Recurrences

All of the recursions (2.1.1), ..., (2.1.4) are linear. Various other important recurrences are non-linear.

Example 2.1.6: The *Catalan recurrence* (introduced in §1.2 and solved in §4.4)

$$\begin{aligned} c_0 &= 1; \\ c_n &= c_0c_{n-1} + c_1c_{n-2} + \cdots + c_{n-1}c_0 \end{aligned}$$

is a homogenous non-linear recurrence without finite degree, with constant coefficients.

2.2 FINDING AN OGF

This section is devoted to the fundamental method for solving a recurrence of the form

$$\begin{aligned} g_0 &= b_0, \ \ldots, \ g_k = b_k; && \text{initial conditions} \\ g_n &= \gamma\,(g_{n-1}, \ldots, g_0) \ \text{for } n > k && \text{recursion} \end{aligned}$$

It uses **three steps** to determine a closed form for the corresponding generating function

$$G(z) = \sum_{n=0}^{\infty} g_n z^n \tag{2.2.1}$$

and then a **fourth step** to derive a closed formula for the coefficients g_n. We describe the four steps of this fundamental method with reference to this **running example** of a recurrence system.

Example 2.2.1: This is a linear homogenous recursion of degree 2 with constant coefficients.

$$\begin{aligned} g_0 &= 1, \quad g_1 = 2; \\ g_n &= 5g_{n-1} - 6g_{n-2} \quad \text{for } n > 1 \end{aligned}$$

Step 1a. Multiply both sides of the recursion equation by z^n.

$$g_n z^n = 5g_{n-1}z^n - 6g_{n-2}z^n$$

Step 1b. Sum both sides of the resulting equation over the same range of values, with a lower bound as low as possible, and upward to ∞.

$$(1b) \qquad \sum_{n=2}^{\infty} g_n z^n \;=\; \sum_{n=2}^{\infty} 5g_{n-1} z^n \;-\; \sum_{n=2}^{\infty} 6g_{n-2} z^n$$

We start all the sums at the lower bound $n = 2$, because starting any lower would take the subscript of g_n below 0 on the left side, and this recurrence system does not specify either g_{-2} or g_{-1}.

Step 2. Recalling equation (2.2.1), we observe that

$$\sum_{n=2}^{\infty} g_n z^n \;=\; \sum_{n=0}^{\infty} g_n z^n \;-\; g_1 z \;-\; g_0 \;=\; G(z) \;-\; g_1 z \;-\; g_0$$

Thus, we can **replace each infinite sum** in equation (1b) by an algebraic expression involving the generating function $G(z)$.

$$(2a) \quad \sum_{n=2}^{\infty} g_n z^n \;=\; 5z \sum_{n=2}^{\infty} g_{n-1} z^{n-1} \;-\; 6z^2 \sum_{n=2}^{\infty} g_{n-2} z^{n-2}$$

$$(2b) \quad G(z) - g_1 z - g_0 \;=\; 5z\,(G(z) - g_0) \;-\; 6z^2 G(z)$$

In $(2a)$, we factor the terms of each sum on the right, so that the power of z in the summand equals the subscript. In $(2b)$, we replace all three infinite sums.

Step 3. Solve for $G(z)$.

$$(3a) \qquad \begin{aligned} G(z)\left(1-5z+6z^2\right) &= g_1 z + g_0 - 5g_0 z \\ &= 2z + 1 - 5z \end{aligned}$$

$$(3b) \qquad G(z) = \frac{1-3z}{1-5z+6z^2}$$

In $(3a)$ we collect the $G(z)$ terms on the left and substitute initial values for the low-subscripted entries of the sequence. In $(3b)$, we isolate the generating function $G(z)$ on the left.

Step 4. Solve for g_n.

$$(4a) \qquad G(z) = \frac{1-3z}{(1-2z)(1-3z)} = \frac{1}{1-2z}$$

$$(4b) \qquad = \sum_{n=0}^{\infty} 2^n z^n \qquad \Rightarrow \qquad g_n = 2^n$$

Step $(4a)$ converts the result of step $(3b)$ into a more tractable form. In $(4b)$ we extract the coefficient g_n.

Check the Answer: A better way to confirm the answer than by retracing the steps is to verify that the answer $g_n = 2^n$ satisfies the recurrence.

$$\begin{aligned} g_0 &= 2^0 = 1, \quad g_1 = 2^1 = 2; && \text{initial conditions} \\ g_n &= 5g_{n-1} - 6g_{n-2} && \text{recursion} \\ &= 5 \cdot 2^{n-1} - 6 \cdot 2^{n-2} \\ &= 5 \cdot 2^{n-1} - 3 \cdot 2^{n-1} \\ &= 2 \cdot 2^{n-1} \\ &= 2^n \end{aligned}$$

Step $(4a)$ is usually not quite this simple, as illustrated by this variation on the running example.

Example 2.2.1, cont.: Suppose that the initial values in the preceding problem were changed to

$$g_0 = 0,\ g_1 = 2$$

Then steps 1 and 2 would be as before. However, here is how we would finish in the modified problem.

Step 3. Solve for $G(z)$.

$$\begin{aligned} (3a) \qquad G(z)(1-5z+6z^2) &= g_1 z + g_0 - 5zg_0 \\ &= 2z + 0 - 0 \\ (3b) \qquad G(z) &= \frac{2z}{1-5z+6z^2} \end{aligned}$$

Step $(3a)$ collects the $G(z)$ terms on the left and substitutes initial values for the low-subscripted entries of the sequence. Step $(3b)$ isolates the generating function $G(z)$ on the left.

Step 4. Solve for g_n. Step $(4a)$ anticipates a method called *partial fraction decomposition*, which is described in the next section. For now, we can confirm that the calculation in Step $(4a)$ is correct, by proceeding from right to left on its top line. The next section describes how to do such a calculation from left to right.

$$(4a)\ G(z) = \frac{2z}{(1-2z)(1-3z)} = \frac{-2}{1-2z} + \frac{2}{1-3z}$$

$$= \sum_{n=0}^{\infty} (-2)2^n z^n + \sum_{n=0}^{\infty} 2 \cdot 3^n z^n$$

$$(4b) \qquad \Rightarrow \quad g_n = -2^{n+1} + 2 \cdot 3^n$$

Check the Answer: As before, we verify that the answer satisfies the recurrence. This time the answer is

$$g_n = -2^{n+1} + 2 \cdot 3^n$$

$$\begin{aligned}
g_0 &= -2^1 + 2 \cdot 3^0 = -2 + 2 = 0 \qquad \text{initial conditions} \\
g_1 &= -2^2 + 2 \cdot 3^1 = -4 + 6 = 2 \\
g_n &= 5g_{n-1} - 6g_{n-2} \qquad\qquad\qquad\qquad \text{recursion} \\
&= 5\left(-2^n + 2 \cdot 3^{n-1}\right) - 6\left(-2^{n-1} + 2 \cdot 3^{n-2}\right) \\
&= (-5) \cdot 2^n + 10 \cdot 3^{n-1} + 3 \cdot 2^n - 4 \cdot 3^{n-1} \\
&= -2 \cdot 2^n + 6 \cdot 3^{n-1} \\
&= -2^{n+1} + 2 \cdot 3^n
\end{aligned}$$

Example 2.2.2: The method of generating functions also solves non-homogeneous recurrences. We illustrate this with a revisit to the Hanoi recurrence.

$$h_0 = 0; \qquad h_n = 2h_{n-1} + 1 \quad \text{for } n > 0$$

We proceed through the same four steps.

$$(1a) \qquad h_n z^n = 2h_{n-1} z^n + 1z^n$$

$$(1b) \qquad \sum_{n=1}^{\infty} h_n z^n = \sum_{n=1}^{\infty} 2h_{n-1} z^n + \sum_{n=1}^{\infty} z^n$$

$$(2a) \qquad \sum_{n=1}^{\infty} h_n z^n \;=\; 2z \sum_{n=1}^{\infty} h_{n-1} z^{n-1} \;+\; z \sum_{n=1}^{\infty} z^{n-1}$$

$$(2b) \qquad H(z) \;-\; h_0 \;=\; 2zH(z) \;+\; \frac{z}{1-z}$$

$$(3a) \qquad H(z)(1-2z) \;=\; h_0 + \frac{z}{1-z} \;=\; 0 \;+\; \frac{z}{1-z}$$

$$(3b) \qquad H(z) \;=\; \frac{z}{(1-z)(1-2z)}$$

We explain in §2.3 how to split a rational function.

$$(4a) \quad H(z) \;=\; \frac{1}{1-2z} \;-\; \frac{1}{1-z}$$

$$(4b) \qquad \;=\; \sum_{n=0}^{\infty} 2^n z^n \;-\; \sum_{n=0}^{\infty} z^n \qquad \Rightarrow\; h_n \;=\; 2^n \;-\; 1$$

This solution was suggested in §1.2 by examination of small cases and then confirmed by mathematical induction.

2.3 PARTIAL FRACTIONS

Suppose that a linear recurrence

$$x_n \;=\; a_{n-1}x_{n-1} + a_{n-2}x_{n-2} + \cdots + a_0x_0 \;+\; \alpha(n)$$

has constant coefficients a_j and that its particularity function $\alpha(n)$ is a polynomial in n. Then the generating function constructed by Steps 1, 2, and 3 of the method of §2.2 is a proper rational function.

$$G(z) \;=\; \sum_{n=0}^{\infty} g_n z^n \;=\; \frac{b_0 + b_1 z + \cdots + b_s z^s}{c_0 + c_1 z + \cdots + c_t z^t}$$

Step 4 is to complete the solution, by deriving a closed formula for g_n. This section develops the details of Step 4. Like the previous section, this section explains the details of the method with the aid of a running example.

Example 2.3.1: The running example now is the rational function

$$G(z) \;=\; \frac{1 - 5z}{1 - 7z + 16z^2 - 12z^3}$$

One may verify that it corresponds to the recurrence

$$\begin{aligned} g_0 &= 1, \quad g_1 = 2, \quad g_2 = -2; \\ g_n &= 7g_{n-1} - 16g_{n-2} + 12g_{n-3} \quad \text{for } n > 2 \end{aligned}$$

Step 4a-1. Factor the denominator into linear factors.

$$c_0 + c_1 z + \cdots + c_t z^t = c_0(1-\tau_1 z)^{\varepsilon_1} \cdots (1-\tau_k z)^{\varepsilon_k}$$

with $\varepsilon_1 + \cdots + \varepsilon_k = t$. For simplicity, we take $c_0 = 1$. For our example, we have

$$\begin{aligned} G(z) &= \frac{1-5z}{1-7z+16z^2-12z^3} \\ &= \frac{1-5z}{(1-2z)^2(1-3z)} \end{aligned}$$

By what is called the *Fundamental Theorem of Algebra*, a polynomial with complex coefficients has a factorization into powers of linear polynomials.

Remark: There is no general method for calculating the roots of a polynomial exactly for higher degree polynomials. Nonetheless, in practice, one commonly encounters polynomials that can be factored by elementary methods.

Step 4a-2. Analyze the rational function into a sum of k rational functions, each of whose denominators is one of the factors $(1-\tau_j)^{\varepsilon_j}$, and whose numerators are "unknown polynomials", each of the respective form

$$b_{j,0} + b_{j,1} z + \cdots + b_{j,\varepsilon_j - 1} z^{\varepsilon_j - 1}$$

Thus,

$$\frac{1-5z}{1-7z+16z^2-12z^3} = \frac{b_{1,0}+b_{1,1}z}{(1-2z)^2} + \frac{b_{2,0}}{(1-3z)}$$

Step 4a-3. Recombine these summands, with a single denominator. For the present example,

$$\frac{1-5z}{1-7z+16z^2-12z^3} = \frac{(b_{1,0}+b_{1,1}z)(1-3z)+b_{2,0}(1-2z)^2}{(1-2z)^2(1-3z)}$$

Step 4a-4. Then collect terms according to the exponent of the factor z^ℓ. For the present example,

$$= \frac{(b_{1,0}+b_{2,0})+(-3b_{1,0}+b_{1,1}-4b_{2,0})z+(-3b_{1,1}+4b_{2,0})z^2}{1-7z+16z^2-12z^3}$$

Step 4a-5. Next obtain a system of t linear equations in t unknowns $b_{j,i}$ by equating each resulting coefficient of z^ℓ in the numerator to the corresponding coefficient of z^ℓ in the numerator of the original linear function, and solve that system.

$$\left.\begin{array}{rcl} b_{1,0} + b_{2,0} &=& 1 \\ -3b_{1,0}+b_{1,1}-4b_{2,0} &=& -5 \\ -3b_{1,1}+4b_{2,0} &=& 0 \end{array}\right\} \Rightarrow \begin{array}{rcl} b_{1,0} &=& 7 \\ b_{1,1} &=& -8 \\ b_{2,0} &=& -6 \end{array}$$

Step 4a-6. Now substitute these solutions into the right side of the equation of Step 4a-2.

$$\frac{1-5z}{1-7z+16z^2-12z^3} = \frac{7-8z}{(1-2z)^2} + \frac{-6}{1-3z}$$

Step 4a-7. Transform each term on the right into the product of its numerator with the power series corresponding, via Corollary 1.7.5, to its denominator. Then simplify

each power series.

$$= (7-8z)\sum_{n=0}^{\infty}\binom{n+1}{1}2^n z^n + (-6)\sum_{n=0}^{\infty}\binom{n}{0}3^n z^n$$

$$= (7-8z)\sum_{n=0}^{\infty}(n+1)2^n z^n - 6\sum_{n=0}^{\infty}3^n z^n$$

Step 4a-8. Finish by combining into a single power series, and then extracting a closed formula for g_n.

$$= \sum_{n=0}^{\infty}[(3n+7)\cdot 2^n - 6\cdot 3^n] \;\Rightarrow\; g_n = (3n+7)\cdot 2^n - 6\cdot 3^n$$

2.4 CHARACTERISTIC ROOTS

The ***method of characteristic roots*** assumes that a homogenous linear recurrence of fixed degree d with constant coefficients has solutions of the form

$$g_n \;=\; \tau_j^{\,n}$$

Example 2.4.1: Running example for this section.

$$\begin{aligned} g_0 &= 1, \quad g_1 = 2; \\ g_n &= 5g_{n-1} - 6g_{n-2} \quad \text{for } n > 1 \end{aligned}$$

Characteristic Equation

Step 1. Form the ***characteristic equation***, as follows.

(1a) Substitute τ^n for g_n in the recurrence.

$$\tau^n \;=\; 5\tau^{n-1} - 6\tau^{n-2}$$

(1b) Factor out τ^{n-d}.

$$\tau^2 \;=\; 5\tau - 6$$

(1c) Move the non-zero terms to the left of the equals sign

$$\tau^2 - 5\tau + 6 \;=\; 0$$

thereby forming the ***characteristic polynomial.***

Step 2. Factor the characteristic polynomial.

$$(\tau - 2)(\tau - 3) \;=\; 0$$

We call the roots of the characteristic polynomial

$$\tau_1 \;=\; 2 \quad \text{and} \quad \tau_2 \;=\; 3$$

characteristic roots. They correspond to the linear factors of the denominator of the generating function derived in Step 4 of Example 2.2.1. We observe that

$$g_n \;=\; \tau_1^n \;=\; 2^n \quad \text{and} \quad g_n \;=\; \tau_2^n \;=\; 3^n$$

are solutions to the given recurrence.

Step 3. As a general solution to the given homogeneous recurrence, form a linear combination of the characteristic roots, using unknown coefficients. If none of the roots is repeated, the result of this step is as follows.

$$g_n \;=\; b_1 2^n + b_2 3^n$$

We shall eventually return to this step to elaborate on the case in which one or more roots is repeated.

Step 4a. Use the initial conditions to write a system of linear equations for the unknown coefficients.

$$\begin{aligned} g_0 \;&=\; 1 \;=\; b_1 2^0 + b_2 3^0 \;=\; b_1 + b_2 \\ g_1 \;&=\; 2 \;=\; b_1 2^1 + b_2 3^1 \;=\; 2b_1 + 3b_2 \end{aligned}$$

Step 4b. Solve for the unknown coefficients.

$$b_1 = 1 \qquad b_2 = 0$$

Step 4c. Substitute the solutions from Step 4b into the general solution of Step 3.

$$g_n = 2^n$$

We observe that this is the same solution previously obtained for this recurrence in Example 2.2.1.

Alternative Initial Values

Suppose that we now consider, as in the continuation of Example 2.2.1, the alternative initial values

$$g_0 = 0, \quad g_1 = 2$$

Then the finish would be as follows.

Step 4a. Use the initial conditions to write a system of linear equation for the unknown coefficients.

$$\begin{aligned} g_0 &= 0 = b_1 2^0 + b_2 3^0 = b_1 + b_2 \\ g_1 &= 2 = b_1 2^1 + b_2 3^1 = 2b_1 + 3b_2 \end{aligned}$$

Step 4b. Solve for the unknown coefficients.

$$b_1 = -2 \qquad b_2 = 2$$

Step 4c. Substitute the solutions from Step 4b into the general solution of Step 3.

$$g_n \;=\; -2^{n+1} \;+\; 2 \cdot 3^n$$

This is the same solution obtained previously, in Example 2.2.1, with these alternative initial values.

Repeated Roots

We now apply the method of characteristic roots to the recurrence of Example 2.3.1.

$$\begin{aligned} g_0 \;&=\; 1, \quad g_1 \;=\; 2, \quad g_2 \;=\; -2; \\ g_n \;&=\; 7g_{n-1} \;-\; 16g_{n-2} \;+\; 12g_{n-3} \quad \text{for } n > 2 \end{aligned}$$

Step 1. The characteristic equation is

$$\tau^3 \;-\; 7\tau^2 \;+\; 16\tau \;-\; 12 \;=\; 0$$

Step 2. Factor the characteristic polynomial.

$$(\tau - 2)^2 \, (\tau - 3) \;=\; 0$$

Step 3. If a root τ_j has multiplicity ε_j, then use

$$b_{j,0} \, \tau_j^{\,n} \;+\; b_{j,1} \, n\tau_j^{\,n} \;+\; \cdots \;+\; b_{j,\varepsilon_j - 1} \, n^{\,\varepsilon_j - 1} \tau_j^{\,n}$$

in forming the general solution with unknown coefficients. In the present example, the general solution is

$$g_n = b_{1,0}2^n + b_{1,1}n2^n + b_2 3^n$$

Step 4a. Use the initial conditions to write a system of linear equation for the unknown coefficients.

$$\begin{aligned} g_0 = 1 &= b_{1,0}2^0 + b_{1,1}0 \cdot 2^0 + b_2 3^0 = b_{1,0} + b_2 \\ g_1 = 2 &= b_{1,0}2^1 + b_{1,1}1 \cdot 2^1 + b_2 3^1 = 2b_{1,0} + 2b_{1,1} + 3b_2 \\ g_2 = -2 &= b_{1,0}2^2 + b_{1,1}2 \cdot 2^2 + b_2 3^2 = 4b_{1,0} + 8b_{1,1} + 9b_2 \end{aligned}$$

Step 4b. Solve for the unknown coefficients.

$$b_{1,0} = 7 \quad b_{1,1} = 3 \quad b_2 = -6$$

Step 4c. Substitute the solutions from Step 4b into the general solution of Step 3.

$$g_n = 7 \cdot 2^n + 3n \cdot 2^n - 6 \cdot 3^n$$

This is the same solution obtained in Example 2.3.1.

Remark: The proof that this method works is a matter of checking that it always yields the same solution as the method of generating functions.

Non-homogeneous Equations

To extend the method of characteristic roots to a non-homogeneous linear recurrence of degree d with constant coefficients

$$\begin{aligned} g_0 &= b_0, \ \ldots, \ g_{d-1} = b_{d-1}; \\ g_n &= a_{n-1}\, g_{n-1} + \cdots + a_{n-d}\, g_{n-d} + \alpha(n) \end{aligned}$$

we first isolate the ***associated homogeneous recurrence***

$$\hat{g}_n \;=\; a_{n-1}\,\hat{g}_{n-1} + \;\cdots\; + a_{n-d}\,\hat{g}_{n-d}$$

obtained by dropping the particularity function.

We illustrate the rest of the extended method with a revisit to the Hanoi recurrence.

$$\begin{aligned} h_0 &= 0; \\ h_n &= 2h_{n-1} + 1 \quad \text{for } n > 0 \end{aligned}$$

Use Steps 1, 2, and 3 to find a general solution to the homogeneous recurrence $\hat{h}_n - 2\hat{h}_{n-1} = 0$.

Steps 1, 2. $\tau - 2 = 0$

Step 3. $\hat{h}_n = b \cdot 2^n$

The result so far is called the ***homogeneous part*** of the general solution.

Step 3N. Find a trial function $\dot{h}_n$ that satisfies the original recurrence. It is called the ***particular solution*** or the ***particular part***. It usually resembles the particularity function. For instance, if the particularity function is a polynomial in n, then the trial function can be a polynomial of the same degree, with unknown coefficient. Since the particularity function for the Hanoi recurrence is a constant, the trial function can be a constant.

$$\dot{h}_n \;=\; c$$

Substitution into the original recurrence leads to a system of linear equations in the unknown coefficients.

$$\begin{aligned} \dot{h}_n &= 2\dot{h}_{n-1} + 1 && \text{(recurrence)} \\ c &= 2c + 1 && \text{(after substitution)} \\ c &= -1 && \text{(particular solution)} \\ h_n &= \hat{h}_n + \dot{h}_n = b \cdot 2^n - 1 && \text{(general solution)} \end{aligned}$$

Step 4a. Use the initial conditions to write a system of linear equations for the unknown coefficients.

$$h_0 = 0 = b \cdot 2^0 - 1 = b - 1$$

Step 4b. Solve for the unknown coefficients.

$$b = 1$$

Step 4c. Substitute the solutions for b_1 and b_2 from Step 4b into the general solution of Step 3.

$$h_n = 2^n - 1$$

This is the same solution obtained in Example 2.2.2 by the method of generating functions.

Example 2.4.2: We modify Example 2.4.1 by giving the recurrence a polynomial particularity function

$$\begin{aligned} g_0 &= 1, \quad g_1 = 2; \\ g_n &= 5g_{n-1} - 6g_{n-2} + 4n - 3 \quad \text{for } n > 1 \end{aligned}$$

We have previously derived for the homogeneous recurrence the general solution

$$\hat{g}_n \;=\; b_1 2^n + b_2 3^n$$

Step 3N. As a particular solution we use the form

$$\dot{g}_n \;=\; c_1 n + c_0$$

and substitute it into the particularized recurrence.

$$\begin{aligned} 4n-3 &= \dot{g}_n - 5\dot{g}_{n-1} + 6\dot{g}_{n-2} && \text{(recurrence)} \\ &= (c_1 n + c_0) - 5(c_1(n-1)+c_0) + 6(c_1(n-2)+c_0) \\ &= n(c_1 - 5c_1 + 6c_1) + (c_0 - 5c_1 + 5c_0 - 12c_1 + 6c_0) \\ &= 2nc_1 + 2c_0 - 7c_1 && \text{(after substituting)} \end{aligned}$$

This leads to the linear equations and solutions

$$\left.\begin{aligned} 4 &= 2c_1 \\ -3 &= 2c_0 - 7c_1 \end{aligned}\right\} \qquad c_1 = 2 \quad c_0 = \frac{11}{2}$$

which are combined with the general solution to the homogeneous part.

$$g_n \;=\; \hat{g}_n + \dot{g}_n \;=\; b_1 2^n + b_2 3^n + 2n + \frac{11}{2}$$

Step 4a. Use the initial conditions to write a system of linear equations for the unknowns b_1 and b_2.

$$\begin{aligned} g_0 &= 1 = b_1 2^0 + b_2 3^0 + \frac{11}{2} \\ g_1 &= 2 = b_1 2^1 + b_2 3^1 + 2\cdot 1 + \frac{11}{2} \end{aligned}$$

Step 4b. Solve for the unknowns b_1 and b_2.

$$\left.\begin{aligned} -\frac{9}{2} &= b_1 + b_2 \\ -\frac{11}{2} &= 2b_1 + 3b_2 \end{aligned}\right\} \qquad \begin{aligned} b_1 &= -8 \\ b_2 &= \frac{7}{2} \end{aligned}$$

Step 4c. Substitute the solutions for b_1 and b_2 from Step 4b into the general solution from Step 3N.

$$g_n = -8 \cdot 2^n + \frac{7}{2} \cdot 3^n + 2n + \frac{11}{2}$$

Example 2.4.3: We now modify Example 2.4.1 by giving the recurrence an exponential particularity function.

$$\begin{aligned} g_0 &= 1, \quad g_1 = 2; \\ g_n &= 5g_{n-1} - 6g_{n-2} + (-1)^n \quad \text{for } n > 1 \end{aligned}$$

We have previously derived for the homogeneous recurrence, as in Example 2.4.2, the general solution

$$\hat{g}_n = b_1 2^n + b_2 3^n$$

Step 3N. As a particular solution we use the form

$$\dot{g}_n = c(-1)^n$$

and substitute it into the particularized recurrence.

$$\begin{aligned} (-1)^n &= \dot{g}_n - 5\dot{g}_{n-1} + 6\dot{g}_{n-2} && \text{(recurrence)} \\ &= c\,(-1)^n + 5c\,(-1)^n + 6c\,(-1)^n \\ &= 12c\,(-1)^n && \text{(after substituting)} \\ c &= \frac{1}{12} && \text{(solution)} \end{aligned}$$

Combine this solution with the general solution to the homogeneous part.

$$g_n = \hat{g}_n + \dot{g}_n = b_1 2^n + b_2 3^n + \frac{1}{12}(-1)^n$$

Step 4a. Use the initial conditions to write a system of linear equations for the unknowns b_1 and b_2.

$$\begin{aligned} g_0 &= 1 = b_1 2^0 + b_2 3^0 + \frac{1}{12} \\ g_1 &= 2 = b_1 2^1 + b_2 3^1 - \frac{1}{12} \end{aligned}$$

Step 4b. Solve for the unknowns b_1 and b_2.

$$\left.\begin{aligned} \frac{11}{12} &= b_1 + b_2 \\ \frac{25}{12} &= 2b_1 + 3b_2 \end{aligned}\right\} \qquad \begin{aligned} b_1 &= \frac{8}{12} \\ b_2 &= \frac{3}{12} \end{aligned}$$

Step 4c. Substitute the solutions for b_1 and b_2 from Step 4b into the general solution from Step 3N.

$$g_n = \frac{8}{12} \cdot 2^n + \frac{3}{12} \cdot 3^n + \frac{1}{12}(-1)^n$$

Complex Roots

A recurrence in which the initial values are real and the recursion has real coefficients has a characteristic polynomial with real coefficients. The roots of such a polynomial may be complex.

Example 2.4.4: The recurrence

$$\begin{aligned} g_0 &= 1, \quad g_1 = 2; \\ g_n &= 2g_{n-1} - 2g_{n-2} \quad \text{for } n > 1 \end{aligned}$$

has the characteristic equation

$$\tau^2 - 2\tau + 2 = 0$$

with roots

$$\tau_1 = 1 + i \quad \text{and} \quad \tau_2 = 1 - i$$

Thus, the general solution is

$$g_n = b_1(1+i)^n + b_2(1-i)^n$$

The init conds yield the complex simultaneous equations

$$\begin{aligned} g_0 &= 1 = b_1(1+i)^0 + b_2(1-i)^0 = b_1 + b_2 \\ g_1 &= 2 = b_1(1+i)^1 + b_2(1-i)^1 \end{aligned}$$

with solution

$$b_1 = \frac{i+1}{2i} \qquad b_2 = \frac{i-1}{2i}$$

Hence, the general solution is

$$g_n = \frac{1}{2i}(1+i)^{n+1} - \frac{1}{2i}(1-i)^{n+1}$$

2.5 SIMULTANEOUS RECURSIONS

With simultaneous recurrences, one uses a substitution from one recursion to reduce the number of different sequences occurring in other recursions. The objective is to reduce the solution of the initial system to the solution of one or more independent linear recurrences.

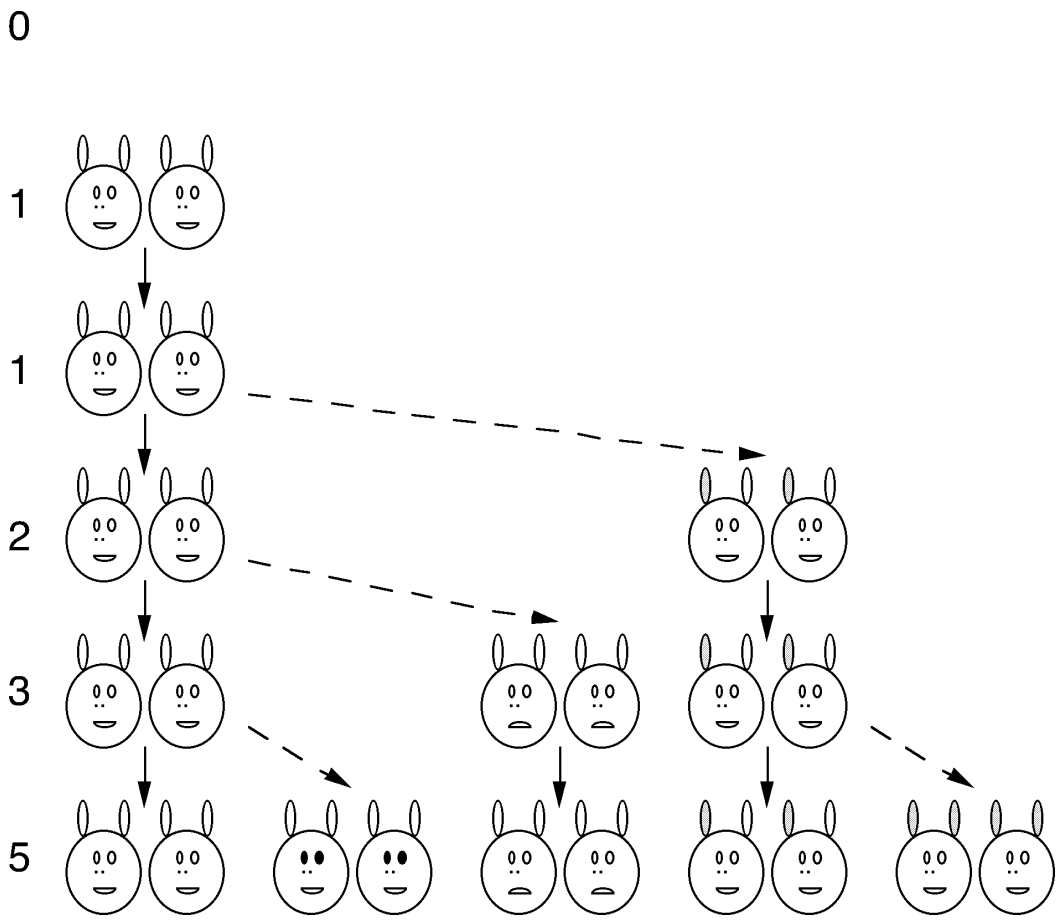


Fig 2.5.1 **Fibonacci rabbit population growth.**

Fibonacci Rabbits

In 1202, Fibonacci imagined a kind of rabbit that takes one month from birth to mature, with a gestation period of one month. Every mature female gives birth each month to a litter of two, with one male and one female. Let b_n represent the number of pairs of newborn rabbits, and let a_n be the number of pairs of adult (mature) rabbits. Suppose that there are no rabbits at $n = 0$ months, and that a newborn pair initiates the system after 1 month.

The total number $f_n = a_n + b_n$ pairs of rabbits is modeled by a ***simultaneous recursion*** with initial conditions

$$a_0 = 0, \quad a_1 = 0, \quad b_0 = 0, \quad b_1 = 1;$$

and the relational equations

$$\begin{aligned} a_n &= a_{n-1} + b_{n-1} \\ b_n &= a_{n-1} \\ f_n &= a_n + b_n \end{aligned}$$

A first step in solving is to use substitutions to reduce it to a recurrence with a single unknown. We see that

$$a_n = a_{n-1} + b_{n-1} = f_{n-1} \tag{2.5.1}$$

$$b_n = a_{n-1} = f_{n-2} \tag{2.5.2}$$

$$f_n = a_n + b_n = f_{n-1} + f_{n-2} \tag{2.5.3}$$

and that $f_0 = a_0 + b_0 = 0$ and $f_1 = a_1 + b_1 = 0 + 1 = 1$.

The resulting single-variable recurrence

$$\begin{aligned} f_0 &= 0,\ f_1 = 1 \\ f_n &= f_{n-1} + f_{n-2} \end{aligned}$$

is recognizable as the ***Fibonacci recurrence.***

Ubiquitousness of the Fibonacci Seq

Although Fibonacci's rabbit model is Fibonacci's invention, the sequence it yields is evidently nature's invention. For instance, what follows immediately is an explanation of an occurrence of the Fibonacci sequence in the construction of a nautilus shell.

DEF: A ***Fibonacci rectangle*** is any rectangle, subdivided into squares whose sides are of lengths that are Fibonacci numbers, in the following sequence:

- The Fibonacci rectangle r_1 is a 1×1 square.
- For each $n \geq 2$, the Fibonacci rectangle r_n is constructed by placing a square along the longer side of the rectangle r_{n-1}, as in Figure 2.5.2.

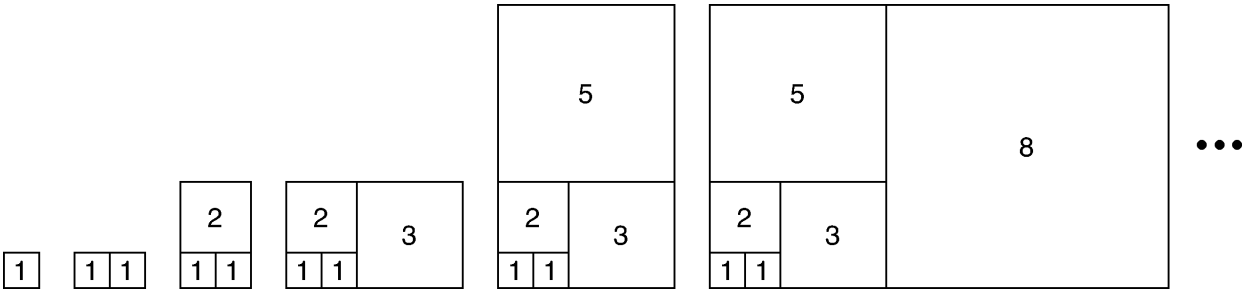


Fig 2.5.2 Fibonacci rectangles.

DEF: A ***spiraled Fibonacci rectangle*** is a Fibonacci rectangle in which each square of size 5×5 and larger is placed so that it touches three previous squares, rather than two. Figure 2.5.3 illustrates a spiraled Fibonacci rectangle.

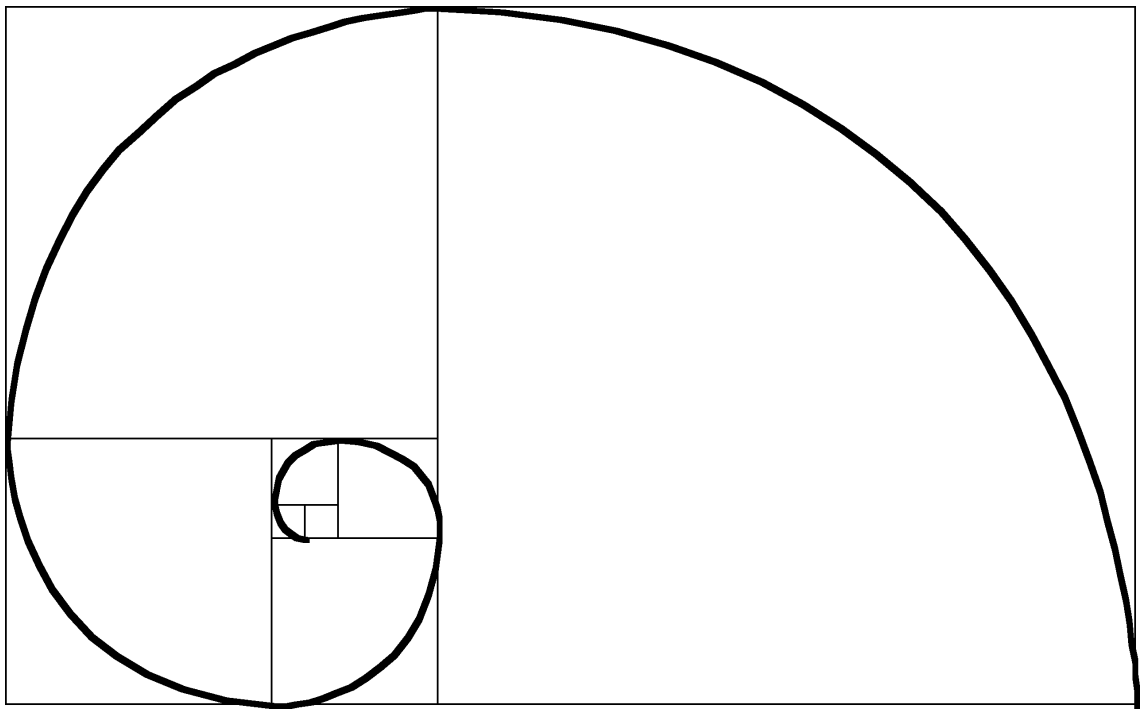


Fig 2.5.3 Fibonacci spiral.

We observe that the inscribed spiral in Figure 2.5.3 has the shape of a nautilus shell. It is called a *Fibonacci spiral.*

Solving the Fibonacci Recurrence

We now use the method of generating functions to solve the Fibonacci recurrence.

Step 1. $f_n z^n = f_{n-1} z^n + f_{n-2} z^n$.

$$\sum_{n=2}^{\infty} f_n z^n = \sum_{n=2}^{\infty} f_{n-1} z^n + \sum_{n=2}^{\infty} f_{n-2} z^n$$

Step 2. Use $F(z)$ as the generating function for f_n.

$$\sum_{n=2}^{\infty} f_n z^n = z \sum_{n=2}^{\infty} f_{n-1} z^{n-1} + z^2 \sum_{n=2}^{\infty} f_{n-2} z^{n-2}$$

$$F(z) - f_1 z - f_0 = z\,(F(z) - f_0) + z^2 F(z)$$

Step 3. Solve for $F(z)$.

$$F(z)(1 - z - z^2) = f_1 z + f_0 - f_0 z = 1z + 0 - 0z = z$$

$$F(z) = \frac{z}{1 - z - z^2}$$

Step 4. To solve for f_n, we use the quadratic equation

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

whose roots involve the *golden mean* and its conjugate

$$\gamma = \frac{1+\sqrt{5}}{2} \quad \text{and} \quad \hat{\gamma} = \frac{1-\sqrt{5}}{2}$$

respectively. We then use partial fractions

$$F(z) = \frac{z}{1 - z - z^2} = \frac{1}{\sqrt{5}} \cdot \left(\frac{1}{1 - \gamma z} - \frac{1}{1 - \hat{\gamma} z}\right)$$

from which we conclude

$$f_n = \frac{1}{\sqrt{5}} \cdot (\gamma^n - \hat{\gamma}^n) \tag{2.5.4}$$

which is called the ***Binet formula*** for the Fibonacci numbers, after Jacquet Binet, who rediscovered it in 1843, after Euler had published it in 1765. Closed forms for a_n and b_n are readily derivable from (2.5.1) and (2.5.2), respectively.

Proposition 2.5.1. *The Fibonacci number f_n is asymptotic to $\frac{\gamma^n}{\sqrt{5}}$.*

Proof: Since $\hat{\gamma} < 1$, it follows that $\hat{\gamma}^n$ is asymptotic to 0. Accordingly, using Eq. (2.5.4) above,

$$\begin{aligned} \lim_{n\to\infty} \frac{f_n}{\gamma^n/\sqrt{5}} &= \lim_{n\to\infty} \frac{(\gamma^n + \hat{\gamma}^n)/\sqrt{5}}{\gamma^n/\sqrt{5}} \\ &= \lim_{n\to\infty} 1 + \frac{\hat{\gamma}^n}{\gamma^n} = 1 \end{aligned}$$

♢

Some Tiling Problems

One of the many other contexts in which Fibonacci numbers arise is tiling problems. We visualize paving a $1 \times n$ chessboard with tiles of various lengths. A $1 \times d$ tile is called a d-tile.

Example 2.5.1: Let t_n be the number of ways to cover a $1 \times n$ chessboard with 1-tiles and 2-tiles. We have $t_0 = 1$, which represents covering a degenerate board with the empty arrangement. Figure 2.5.4 shows the possibilities for $n = 0, \ldots, 4$.

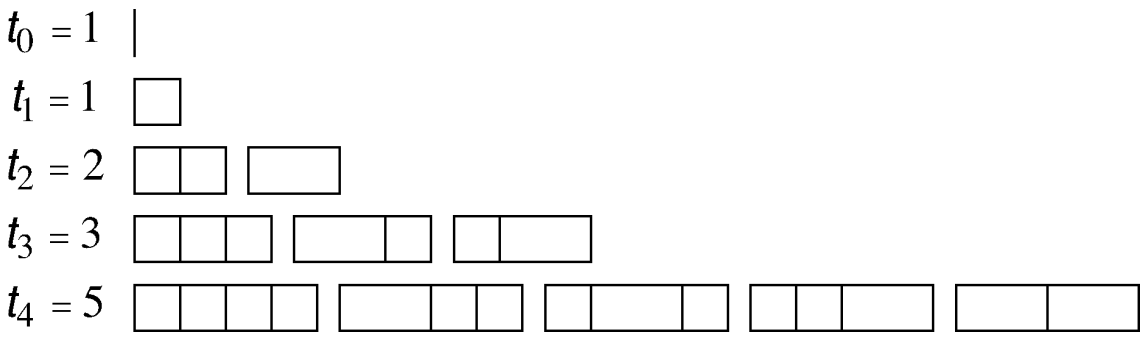


Fig 2.5.4 Tiling a $1 \times n$ chessboard.

The number of $1 \times n$ tilings in which the rightmost tile is a 1-tile is t_{n-1}. The number of $1 \times n$ tilings in which the rightmost tile is a 2-tile is t_{n-2}. The solution to the resulting recurrence

$$\begin{aligned} t_0 &= 1, \quad t_1 = 1; \\ t_n &= t_{n-1} + t_{n-2} \end{aligned}$$

is clearly $t_n = f_{n+1}$, the $n+1^{\text{st}}$ Fibonacci number.

Example 2.5.2: Observe that any tiling in which all the tiles are of odd length can be converted to a tiling with 1-tiles and 2-tiles, whose initial tile is a 1-tile, by breaking a tile of length $2n + 1$ into a 1-tile, followed by n 2-tiles. This breakage operation can be inverted, since each maximal string of 2-tiles and the 1-tile that precedes it can be assembled into an odd-length tile. It follows that there is a one-to-one, onto correspondence between the two kinds of tiling. Since the number of tilings of a $1 \times n$ chessboard with 1-tiles and 2-tiles, and with an initial 1-tile, is the Fibonacci number $t_{n-1} = f_{n-2}$, this must also be the number of tilings with tiles of odd length.

2.6 FIBO NUMBER IDENTITIES

In the first few entries of the Fibonacci sequence

n	0	1	2	3	4	5	6	7	8	9	10	11	12
f_n	0	1	1	2	3	5	8	13	21	34	55	89	144

for each instance of a number n and a multiple mn, the Fibonacci number f_{mn} is a multiple of f_n. For instance,

$$f_5 = 5 \quad \text{and} \quad f_{10} = 55 = 11f_5$$

This section verifies this and other visible patterns.

Forward-Shift and Subscript Multipliers

As a preliminary, we consider a relationship between f_{k+n} and f_n. Proceeding from $k = 2$,

$$\begin{aligned}
f_{n+2} &= f_{n+1} + f_n \\
f_{n+3} &= f_{n+2} + f_{n+1} = (f_{n+1} + f_n) + f_{n+1} \\
&= 2f_{n+1} + f_n \\
f_{n+4} &= f_{n+3} + f_{n+2} = (2f_{n+1} + f_n) + (f_{n+1} + f_n) \\
&= 3f_{n+1} + 2f_n \\
f_{n+5} &= f_{n+4} + f_{n+3} = (3f_{n+1} + 2f_n + (2f_{n+1} + f_n) \\
&= 5f_{n+1} + 3f_n \\
f_{n+6} &= f_{n+5} + f_{n+4} = (5f_{n+1} + 3f_n) + (3f_{n+1} + 2f_n) \\
&= 8f_{n+1} + 5f_n
\end{aligned}$$

we observe that the coefficients of f_{n+1} and f_n are themselves Fibonacci numbers. The observable pattern is confirmed by the following theorem.

Thm 2.6.1 [*Forward-Shift Identity*]. *The Fibonacci numbers satisfy the equation*

$$f_{n+k} \;=\; f_k f_{n+1} + f_{k-1} f_n \qquad \textit{for all } k \geq 1$$

Proof: By induction on k.

BASIS: If $k = 1$, then $f_1 = 1$ and $f_0 = 0$, and, thus,

$$f_k f_{n+1} \;+\; f_{k-1} f_n \;=\; 1 \cdot f_{n+1} \;+\; 0 \cdot f_n \;=\; f_{n+1}$$

IND HYP: Assume for all j in the interval $0 < j < k$ that

$$f_{n+j} \;=\; f_j f_{n+1} \;+\; f_{j-1} f_n$$

IND STEP: Then

$$\begin{aligned}
f_{n+k} &= f_{n+k-1} + f_{n+k-2} && \text{(Fibonacci recursion)} \\
&= (f_{k-1} f_{n+1} + f_{k-2} f_n) + (f_{k-2} f_{n+1} + f_{k-3} f_n) && \text{(ind hyp)} \\
&= (f_{k-1} + f_{k-2}) f_{n+1} + (f_{k-2} + f_{k-3}) f_n && \text{(regrouping)} \\
&= f_k f_{n+1} + f_{k-1} f_n && \text{(Fibonacci recursion)} \quad \diamondsuit
\end{aligned}$$

We now confirm the initial observation regarding multiples.

Cor 2.6.2. *For all $k \geq 0$, the Fibonacci number f_{kn} is a multiple of the Fibonacci number f_n.*

Proof: By induction on the multiplier k.

BASIS: This is trivial for $k = 0$ and $k = 1$. That is,

$$\begin{aligned} f_{0n} &= 0 = 0f_n \\ f_{1n} &= f_n = 1f_n \end{aligned}$$

IND HYP: Assume that the Fibonacci f_{jn} is a multiple of the Fibonacci number f_n, for all j such that $0 \leq j < k$.

IND STEP: Then

$$\begin{aligned} f_{kn} &= f_{n+(k-1)n} \\ &= f_{(k-1)n}\, f_{n+1} + f_{(k-1)n-1}\, f_n \qquad \text{(by Thm 2.6.1)} \end{aligned}$$

By the inductive hypothesis, there is a number M such that $f_{(k-1)n} = Mf_n$. Thus,

$$\begin{aligned} f_{kn} &= Mf_nf_{n+1} + f_{(k-1)n-1}f_n \\ &= (Mf_{n+1} + f_{(k-1)n-1})\, f_n \end{aligned}$$ ♢

Cassini's Identity

In returning to the early entries of the Fibonacci sequence

n	0	1	2	3	4	5	6	7	8	9	10	11	12
f_n	0	1	1	2	3	5	8	13	21	34	55	89	144

we also observe that the square of each Fibonacci number differs by 1 from the product of the Fibonacci number that follows it and the Fibonacci number that precedes it. For instance,

$$\begin{aligned} f_5 f_3 &= 5 \cdot 2 = 3 \cdot 3 + 1 = f_4 f_4 + 1 \\ f_6 f_4 &= 8 \cdot 3 = 5 \cdot 5 - 1 = f_5 f_5 - 1 \end{aligned}$$

Theorem 2.6.3 [*Cassini's Identity*]. *In the Fibonacci sequence* $\langle f_n \rangle$,

$$f_{n+1} f_{n-1} = f_n^2 + (-1)^n \qquad \text{for } n \geq 1$$

Proof: By induction on n.

BASIS: Confirmation that the identity holds for $n = 1$ is as follows.

$$\begin{aligned} f_2 f_0 &= 1 \cdot 0 = 0 \\ f_1^2 + (-1)^1 &= 1 \cdot 1 - 1 = 0 \end{aligned}$$

IND HYP: Assume that

$$f_{k+1} f_{k-1} = f_k^2 + (-1)^k \qquad \text{for } 1 \leq k < n$$

IND STEP: Then

$$\begin{aligned} f_{n+1} f_{n-1} &= (f_n + f_{n-1}) f_{n-1} && \text{(Fibonacci recurrence)} \\ &= f_n f_{n-1} + f_{n-1}^2 \\ &= f_n f_{n-1} + f_n f_{n-2} - (-1)^{n-1} && \text{(ind hyp)} \\ &= f_n (f_{n-1} + f_{n-2}) + (-1)^n \\ &= f_n^2 + (-1)^n && \text{(Fibonacci recurrence)} \quad \diamondsuit \end{aligned}$$

Fibonacci Number System

It is clear that every non-negative integer is the sum of some Fibonacci numbers, since 1 is a Fibonacci number. The following example adds as requirements non-repetition and non-consecutiveness.

Example 2.6.1: Each of the smallest integers that is not a Fibonacci number is the sum of two or more non-consecutive Fibonacci numbers.

$$\begin{aligned} 4 &= 3+1 & \qquad 10 &= 8+2 \\ 6 &= 5+1 & \qquad 11 &= 8+3 \\ 7 &= 5+2 & \qquad 12 &= 8+3+1 \\ 9 &= 8+1 & \qquad 14 &= 13+1 \end{aligned}$$

Moreover, this property holds for some larger examples.

$$100 = 89+8+3 \qquad 200 = 144+55+1$$

Theorem 2.6.4. *Every non-negative integer n is the sum of distinct non-consecutive Fibonacci numbers.*

Proof: By induction on n.

BASIS: The number $n = 0$ is the sum of the empty set.

IND HYP: Assume for some $n > 0$ that every number less than n is representable as the sum of distinct non-consecutive Fibonacci numbers.

IND STEP: Let f_m be the largest Fibonacci number less than or equal to n. Since $f_{m+1} > n$, it follows from the

Fibonacci recursion that

$$f_{m-1} > n - f_m$$

Thus, when the induction hypothesis is applied to $n - f_m$, the summands are non-consecutive Fibonacci numbers, each less than f_{m-1}. Accordingly, when f_m is included in the set of summands, the members of the resulting set of Fibonacci numbers remain non-consecutive, and their sum is n. ♢

DEF: The ***Fibonacci representation of an integer*** is its expression as a sum of distinct non-consecutive Fibonacci numbers.

2.7 NON-CONSTANT COEFFICIENTS

A good method for solving any recurrence that is not specified as a linear recurrence of fixed degree with constant coefficients is to transform it into such a recurrence. We will apply this strategy to the *quicksort recurrence*.

A Reduction Strategy

Consider this general linear recursion of degree d with variable coefficients.

$$\begin{aligned} f(n)\,x_n \;=\; & c_{n-1}\,f(n-1)\,x_{n-1} + \cdots \\ & + c_{n-d}\,f(n-d)\,x_{n-d} + p(n) \end{aligned} \tag{2.7.1}$$

Substituting $f(n)x_n = y_n$ yields the recursion

$$y_n \;=\; c_{n-1}\,y_{n-1} + \cdots + c_{n-d}\,y_{n-d} + p(n) \tag{2.7.2}$$

which is linear with constant coefficients, and, therefore, is amenable to previously developed methods of solution. A solution

$$y_n = g(n)$$

for the recursion (2.7.2) could be reverse-transformed into a solution

$$x_n = g(n)/f(n)$$

for the recursion (2.7.1).

Example 2.7.1: Consider the recurrence

$$\begin{aligned} x_0 &= 0; \\ x_n &= \frac{2(n-1)}{n}\, x_{n-1} + \frac{1}{n} \end{aligned}$$

Multiplying the recursion by n yields the recursion

$$nx_n = 2(n-1)\, x_{n-1} + 1$$

in the form of recurrence (2.7.1). The substitution

$$nx_n = y_n$$

yields this new recurrence in the form of recurrence (2.7.2).

$$\begin{aligned} y_0 &= 0; \\ y_n &= 2y_{n-1} + 1 \end{aligned}$$

This transformed recurrence is easily solved by the method of generating functions or by the method of characteristic roots. Indeed, if we recognize it as the Hanoi recurrence, we already have this solution for y_n:

$$y_n = 2^n - 1$$

To obtain the solution for x_n, we substitute $y_n/n = x_n$:

$$\begin{aligned} nx_n &= 2^n - 1 \\ \Rightarrow \qquad x_n &= \frac{2^n - 1}{n} \end{aligned}$$

Example 2.7.2: To solve the recurrence

$$\begin{aligned} x_0 &= 0; \\ nx_n &= \left(1 - \frac{1}{n}\right)^{n-1} x_{n-1} + (2n)^{1-n} \qquad \text{for } n \geq 1 \end{aligned}$$

we first multiply the recursion by n^{n-1}, thereby obtaining

$$\begin{aligned} n^n x_n &= (n-1)^{n-1} x_{n-1} + n^{n-1}(2n)^{1-n} \\ \Rightarrow \quad n^n x_n &= (n-1)^{n-1} x_{n-1} + 2^{1-n} \end{aligned}$$

Substituting $n^n x_n = y_n$ yields the recurrence

$$\begin{aligned} y_0 &= 0; \\ y_n &= y_{n-1} + \left(\frac{1}{2}\right)^{n-1} \qquad \text{for } n \geq 1 \end{aligned}$$

This transformed recurrence is easily solved.

$$\begin{aligned} y_n &= 1 + \frac{1}{2} + \frac{1}{4} + \cdots + \frac{1}{2^{n-1}} \\ &= 2 - \frac{1}{2^{n-1}} \end{aligned}$$

By reverse-substituting $y_n = n^n x_n$, we solve the given recurrence.

$$x_n = \frac{1}{n^n} \cdot \left(2 - \frac{1}{2^{n-1}}\right) = \frac{2^n - 1}{n^n 2^{n-1}}$$

Sum in a Recurrence: Quicksort

Beyond the complication of variable coefficients, the ***quicksort recurrence*** has no fixed degree. It involves a long sum of earlier values in the sequence. Another preliminary to applying the methods of the earlier part of this chapter is to transform it into a recurrence of fixed degree.

One signature step of ***quicksort*** is choosing a ***pivot*** entry. The other signature step is to ***tripartition*** the given sequence:

- The *front part* contains every element that is less than the pivot. This part may be empty.
- The *pivot part* contains only the pivot entry itself.
- The *back part* contains every entry not in the other two parts, all the entries that are greater than the pivot, plus any duplicates of the pivot. The back part may be empty.

If the length of a sequence is 0 or 1, then the sequence is deemed to be sorted. Otherwise, it is tripartitioned, and then its front part and its back part are quicksorted. In the implementation represented by the following algorithm, the pivot is selected at random. (This tends to produce pivots whose value is relatively near to the median of the sequence, a fortuitous event that reduces the number of subsequent iterations.) The following algorithm specifies the details of a quicksort.

Algorithm 2.7.1: Quicksort

Input: seq $X = \langle x_j \rangle$; range limits lo, hi
Output: that same sequence in non-decreasing order

if $lo \geq hi$ **then return**
    else $pivot := random(\{lo, \ldots, hi\})$
"tripartition" $\langle x_{lo}, \ldots, x_{hi} \rangle$ into $\langle x_{pivot} \rangle$ plus
    $front := \langle x_j \mid x_j < x_{pivot} \rangle$
    $back := \langle x_j (j \neq \text{pivot}) \mid x_j \geq x_{pivot} \rangle$
$X :=$ concatenate(Qsort($front$), x_{pivot}, Qsort($back$))

Example 2.7.3: Suppose that the given sequence

$$(78 \quad 49 \quad 05 \quad 14 \quad 10 \quad 90 \quad 44 \quad 39 \quad 19 \quad 55)$$

gets initial pivot 39. Then the first tripartition is

$$\overbrace{(05 \quad 14 \quad 10 \quad 19)}^{\text{front part}} \quad \overbrace{(39)_q}^{\text{pivot}} \quad \overbrace{(78 \quad 49 \quad 90 \quad 44 \quad 55)}^{\text{back part}}$$

Subscript q denotes a part that is fully quicksorted. Suppose that at the second stage the pivots chosen in the parts not yet fully quicksorted are 10 and 78. Then the result of the second-stage tripartitioning is

$$((05)_q \quad (10)_q \quad (14 \quad 19)) \quad (39)_q$$
$$((49 \quad 44 \quad 55) \quad (78)_q \quad (90)_q)$$

Suppose that at the third stage the pivots chosen in the parts not yet fully quicksorted are 19 and 49. Then the

result of the third-stage tripartitioning is

$$\begin{array}{c} ((05)\ \ (10)\ \ ((14)\ \ (19)))\ \ \ \ (39) \\ (((44)\ \ (49)\ \ (55))\ \ (78)\ \ (90)) \end{array}$$

at which point all parts are fully quicksorted. Concatenation proceeds level by level to this final result.

$$(05\ \ 10\ \ 14\ \ 19\ \ 39\ \ 44\ \ 49\ \ 55\ \ 78\ \ 90)$$

Analysis of the Time to Quicksort

Let Q_n represent the time needed to quicksort a sequence of length n. This involves the following time expenditures:

1	to select a pivot location
n	to tripartition a seq of length n
Q_k	to quicksort a front part of length k
Q_{n-k-1}	to quicksort the back part of length $n-k-1$

The probability that there are exactly k items smaller than random pivot is

$$\text{pr}\,(k \text{ items } < \text{ pivot}) \ = \ \frac{1}{n}$$

This leads to the following recurrence.

$$Q_0 \ = \ 0$$

$$Q_n = 1 + n + \sum_{k=0}^{n-1} \text{pr}\,(k \text{ items} < \text{pivot}) \cdot [Q_k + Q_{n-k-1}]$$

$$Q_n = 1 + n + \sum_{k=0}^{n-1} \frac{1}{n} \cdot [Q_k + Q_{n-k-1}]$$

$$= 1 + n + \frac{2}{n} \sum_{k=0}^{n-1} Q_k$$

An obstacle to solving the recurrence is the unlimited number of terms in the sum. Often, such a recursion can be transformed into a recursion of fixed degree, by setting up a subtraction of sums.

$$nQ_n = n + n^2 + 2\sum_{k=0}^{n-1} Q_k \tag{2.7.3}$$

$$(n-1)Q_{n-1} = (n-1) + (n-1)^2 + 2\sum_{k=0}^{n-2} Q_k$$

$$= n^2 - n + 2\sum_{k=0}^{n-2} Q_k \tag{2.7.4}$$

Productively, subtracting (2.7.4) from (2.7.3) yields

$$nQ_n - (n-1)Q_{n-1} = 2n + 2Q_{n-1}$$

and, thus,

$$nQ_n = (n+1)Q_{n-1} + 2n$$

which may be rewritten in the form

$$\frac{Q_n}{n+1} = \frac{Q_{n-1}}{n} + \frac{2}{n+1}$$

After making the substitution

$$\frac{Q_n}{n+1} = P_n$$

there is the following transformed recurrence

$$\begin{aligned} P_0 &= 0; \\ P_n &= P_{n-1} + \frac{2}{n+1} \qquad \text{for } n \geq 1 \end{aligned}$$

whose solution is

$$P_n = \sum_{k=1}^{n} \frac{2}{k+1} = 2\sum_{k=1}^{n} \frac{1}{k+1} = 2\sum_{j=2}^{n+1} \frac{1}{j} = 2(H_{n+1} - 1)$$

which is then reverse transformed.

$$\begin{aligned} Q_n &= (n+1)P_n = 2(n+1)(H_{n+1} - 1) \\ &= 2(n+1)\left(H_n + \frac{1}{n+1}\right) - 2(n+1) \\ &= 2(n+1)H_n + 2 - 2(n+1) = 2(n+1)H_n - 2n \end{aligned}$$

Confirming Small Cases

Direct application of the recurrence

$$\begin{aligned} Q_0 &= 0; \\ Q_n &= 1+n+\frac{2}{n}\sum_{k=0}^{n-1} Q_k \end{aligned}$$

yields the small values

$$\begin{aligned} Q_1 &= 1+1+\frac{2}{1}[Q_0] = 2+0 = 2 \\ Q_2 &= 1+2+\frac{2}{2}[Q_0+Q_1] = 3+1\cdot 2 = 5 \\ Q_3 &= 1+3+\frac{2}{3}[Q_0+Q_1+Q_2] = 4+\frac{2}{3}\cdot[2+5] = \frac{26}{3} \end{aligned}$$

Application of the closed formula

$$Q_n = 2(n+1)H_n - 2n$$

yields the small values

$$\begin{aligned} Q_1 &= 2\cdot(1+1)H_1 - 2\cdot 1 = 4\cdot 1 - 2 = 2 \\ Q_2 &= 2\cdot(2+1)H_2 - 2\cdot 2 = 6\cdot\frac{3}{2} - 4 = 5 \\ Q_3 &= 2\cdot(3+1)H_3 - 2\cdot 3 = 8\cdot\frac{11}{6} - 6 = \frac{26}{3} \end{aligned}$$

2.8 DIVIDE-&-CONQUER RELATIONS

A *divide-and-conquer strategy* for solving a problem is to partition it into subproblems, such that the total effort needed to do all the subproblems is significantly less than a direct approach to the original problem.

DEF: A recurrence of the form

$$x_n = c\,x_{n/d} + \alpha(n)$$

is said to be a ***divide-and-conquer recurrence.***

Remark: It represents that each of c subproblems is smaller than the original by a factor of d and in which $\alpha(n)$ is the cost of partitioning and recombining.

Binary Search

The signature step of a *binary search* is that the target key is compared to the key of the middle record. If it precedes the middle key, then the target record cannot be in the second half of the file, so it is inactive for the remainder of the search. Otherwise, the first half goes inactive. This step is then applied recursively to the active half, until there is only one active record remaining.

Example 2.8.1: Suppose we are searching for the target value $y = 74$ in the following list of length 16:

$$X = \begin{Bmatrix} 5 & 18 & 31 & 34 & 35 & 39 & 42 & 47 \\ 51 & 53 & 60 & 74 & 75 & 80 & 81 & 96 \end{Bmatrix}$$

Initially, the entire list is active, with a lower limit location of $lo = 1$ and an upper limit location of $hi = 16$.

In the *first stage*, the middle location is determined to be

$$mid = \left\lceil \frac{lo + hi}{2} \right\rceil = \left\lceil \frac{1 + 16}{2} \right\rceil = 9$$

The target value $y = 74$ is compared with the middle value $x_9 = 51$. Since

$$y = 74 \geq x_9 = 51$$

and since the list is sorted, it follows that the target value $y = 74$, if present in the list, must be in the second half of the list, which becomes the only active sector. Resetting the lower limit to $lo = 9$ achieves the choice of active sector.

In the *second stage*, the middle location of the active sector $x_9, \ldots, x_{16}$ is location

$$mid = \left\lceil \frac{lo + hi}{2} \right\rceil = \left\lceil \frac{9 + 16}{2} \right\rceil = 13$$

The target value $y = 74$ is compared with the middle value $x_{13} = 75$. Since

$$y = 74 \leq x_{13} = 75$$

it follows that the target value $y = 74$, if present in the list, must be in the first half of the active sector, which becomes the new active sector. Resetting the upper limit to $hi = 12$ accomplishes this.

In the *third stage*, the middle location of the active sector $x_9, \ldots, x_{12}$ is location

$$mid = \left\lceil \frac{lo + hi}{2} \right\rceil = \left\lceil \frac{9 + 12}{2} \right\rceil = 11$$

The target $y = 74$ is compared with $x_{11} = 60$. Since

$$y = 74 \geq x_{11} = 60$$

it follows that the target $y = 74$, if present, must be in the 2^{nd} half of the active sector, which becomes the current active sector. Therefore, the lower limit is reset to $lo = 11$.

In the *fourth stage*, the middle location of the active sector x_{11}, x_{12} is location

$$mid = \left\lceil \frac{lo + hi}{2} \right\rceil = \left\lceil \frac{11 + 12}{2} \right\rceil = 12$$

The target $y = 74$ is compared with $x_{12} = 74$. Since

$$y = 74 \geq x_{12} = 74$$

it follows that the target $y = 74$, if present in the list, must be in the 2^{nd} half of the active sector, which becomes the final active sector, as the lower limit is reset to $lo = 12$.

The final active sector has only one item. If it is not the target item, then the target item is not in the original list. If it is the target item, as in this example, then its location is returned as the output of the search.

The following algorithm gives the general rules for a binary search.

Algorithm 2.8.1: Recursive Binary Search (RBS)

Input: a non-decr seq $X = \langle x_j \rangle$; range limits lo, hi;
a target value y
Output: if $y \notin \{x_{lo}, \ldots, x_{hi}\}$ then $*$ ("not found");
else $min\{j \in \{lo, \ldots, hi\} \mid y = x_j\}$

call $RBS(X, lo, hi, y)$

$$output := \begin{cases} lo & \text{if } y = x_{lo} \\ * & \text{if } y \neq x_{lo} \end{cases}$$

Recursive Subroutine $RBS(X, lo, hi, y)$
if $lo = hi$ **then return**
else $mid = \lceil (hi + lo)/2 \rceil$
if $y < x_{mid}$ **then** $hi := mid - 1$ **else** $lo = mid$
call $RBS(X, lo, hi, y)$

Analysis of the Time for a Binary Search

Let b_n be the number of comparisons needed to perform a binary search on an array of size n. Since at each stage, the limits of the active search space within the original sequence are reset to about half their previous range, the value of b_n is represented by the following divide-and-conquer binary-search recurrence:

$$\begin{aligned} b_1 &= 2; \\ b_n &= b_{n/2} + 2 \end{aligned}$$

The substitutions $n = 2^k$ and $b_{2^k} = c_k$ transform this to the recurrence

$$\begin{aligned} c_0 &= 2; \\ c_k &= c_{k-1} + 2 \end{aligned}$$

The solution to the transformed recurrence is evidently

$$c_k = 2k + 2$$

from which it follows (by the inverse substitutions $k = \lg n$ and $c_{\lg n} = b_n$) that the solution to the binary-search recurrence is

$$b_n = 2 \lg n + 2$$

Merging

Mergesort is based on repeated merging. Algorithm 2.8.2 prescribes a process for merging two sorted lists.

Algorithm 2.8.2: Merge

Input: non-decreasing lists L_1 and L_2
Output: a merged non-decr list L, initially empty

while both input lists are non-empty
    move $\min(\text{head}(L_1), \text{head}(L_2))$ from its own list
        to the tail of the output list
    if that transfer makes one list empty **then** transfer
        all the remaining elements of the other list to
        the end of the output list

Example 2.8.2: Suppose that the input lists and output list are initially

$$\begin{array}{lccccc} L_1: & 2 & 14 & 30 & 37 & 55 \\ L_2: & 3 & 36 & 43 & 65 & \\ L: & & & & & \end{array}$$

After two transfers, the lists are

$$\begin{array}{lcccc} L_1: & 14 & 30 & 37 & 55 \\ L_2: & 36 & 43 & 65 & \\ L: & 2 & 3 & & \end{array}$$

After two more transfers, the lists are

$$\begin{array}{lccccc} L_1: & 37 & 55 & & & \\ L_2: & 36 & 43 & 65 & & \\ L: & 2 & 3 & 14 & 30 & \end{array}$$

The final lists are

$$\begin{array}{lccccccccc} L_1: & & & & & & & & & \\ L_2: & & & & & & & & & \\ L: & 2 & 3 & 14 & 30 & 36 & 37 & 43 & 55 & 65 \end{array}$$

The time needed to merge the lists L_1 and L_2 is at worst proportional to the sum of their lengths.

Iterative Mergesort

A *mergesort* is a sort by iterative merging of sublists of the initial list.

Example 2.8.3: Suppose that the list to be sorted is

$$X = [82 \quad 48 \quad 03 \quad 17 \quad 11 \quad 94 \quad 41 \quad 37]$$

which has length 8. From an iterative perspective, this list is initially viewed as a list of 8 files, each of length 1.

$$X_1 = [(82) \quad (48) \quad (03) \quad (17) \quad (11) \quad (94) \quad (41) \quad (37)]$$

The files of length 1 are paired, as follows:

$$X_1' = \begin{bmatrix} ((82) & (48)) & ((03) & (17)) \\ ((11) & (94)) & ((41) & (37)) \end{bmatrix}$$

Merging the two sublists of length 1 within each pair yields this file with 4 sorted subfiles, each of length 2.

$$X_2 = [(48 \quad 82) \quad (03 \quad 17) \quad (11 \quad 94) \quad (37 \quad 41)]$$

The sorted subfiles are paired, as follows.

$$X_2' = [((48 \quad 82) \quad (03 \quad 17)) \quad ((11 \quad 94) \quad (37 \quad 41))]$$

Merging the two sublists of length 2 within each pair yields this file with 2 sorted subfiles, each of length 4.

$$X_3 = [(03 \quad 17 \quad 48 \quad 82) \quad (11 \quad 37 \quad 41 \quad 94)]$$

These two sorted subfiles of length 4 are paired.

$$X_3' = [((03 \quad 17 \quad 48 \quad 82) \quad (11 \quad 37 \quad 41 \quad 94))]$$

Then the two subfiles of length 4 are merged, thus ultimately yielding a fully sorted list of length 8.

$$X = [03 \quad 11 \quad 17 \quad 37 \quad 41 \quad 48 \quad 82 \quad 94]$$

Recursive Mergesort

In a recursive mergesort, the order of merger is a bit different from an iterative mergesort. E.g., the first two sorted sublists of length 2 are merged into a single sublist of length 4 before the rest of the sublists of length 1 are merged into sublists of length 2. Of course, the results are identical. Algorithm 2.8.3 is for recursive mergesort.

Algorithm 2.8.3: Recursive Mergesort

Input: $X = \langle x_1, x_2, \ldots, x_n \rangle$
Output: that same sequence in non-decreasing order

Recursive Subroutine $MerSo(X)$
if $n > 1$ **then**
    $m = \lceil n/2 \rceil$
    $X_1 := \langle x_1, x_2, \ldots, x_m \rangle$
    $X_2 := \langle x_{m+1}, x_2, \ldots, x_n \rangle$
    $X := Merge(X_1, X_2)$

Analysis of the Time or a Mergesort

Let s_n be the number of comparisons needed in a mergesort on an array of size n. The value of s_n is represented by the following divide-and-conquer recurrence:

$$\begin{aligned} s_1 &= 1; \\ s_n &= 2s_{n/2} + n \end{aligned}$$

The substitutions $n = 2^k$ and $s_{2^k} = t_k$ transform this into the recurrence

$$\begin{aligned} t_0 &= 1; \\ t_k &= 2t_{k-1} + 2^k \end{aligned}$$

which we can solve with generating functions.

$$\begin{aligned} \sum_{k=1}^{\infty} z^k t_k &= 2z \sum_{k=1}^{\infty} z^{k-1} t_{k-1} + \sum_{k=1}^{\infty} z^k 2^k \\ T(z) - 1 &= 2zT(z) + \frac{2z}{1-2z} \\ T(z) &= \frac{1}{(1-2z)^2} \\ \Rightarrow \qquad t_k &= (k+1)2^k \end{aligned}$$

Thus, after the inverse substitutions $k = \lg n$ and $t_{\lg n} = s_n$, the solution to the mergesort recurrence is

$$s_n = n \lg n + n$$

What enables the divide-and-conquer strategy of a mergesort to succeed at reducing the work effort, relative to naive forms of sorting, is that merging two sorted lists of equal length together takes less work than a naive sort of the union of the two lists. Naive sorts (e.g., insertion sorts and selection sorts) of n items require $\mathcal{O}(n^2)$ steps.

The Josephus Recurrence

During the Roman occupation of the Judean state, the Romans had trapped 41 Jewish rebels at a fortress called Jotapata. Rather than face likely slavery in Rome or public execution, these patriots made a suicide pact. Proceeding around a circle, every third man was to be killed, until there was only one remaining man, who would then kill himself. Joseph ben Mattiyahu ha-Cohen (who adopted the name Flavius Josephus after going over to the Romans), a survivor of several previous losses to the Romans, calculated what would be the last two positions on the circle whose occupants would remain alive, so that he and a friend could survive.

DEF: The ***Josephus problem*** is to calculate a closed formula for the values of the sequence $J_n^{(k)}$, the position of the last man alive, for a circle of n men in which every k^{th} man is killed.

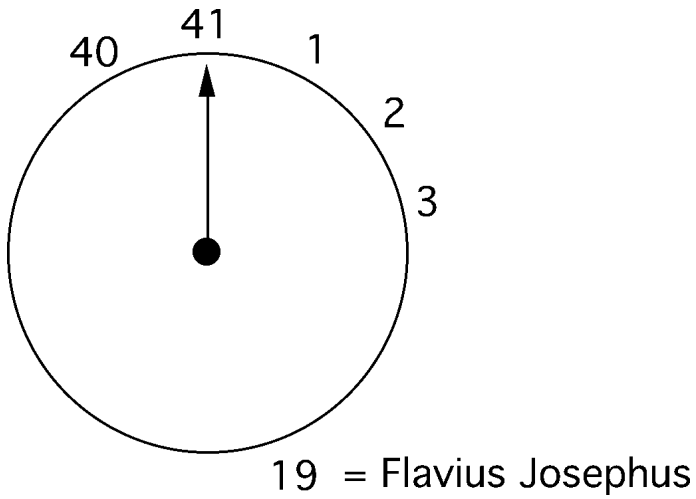


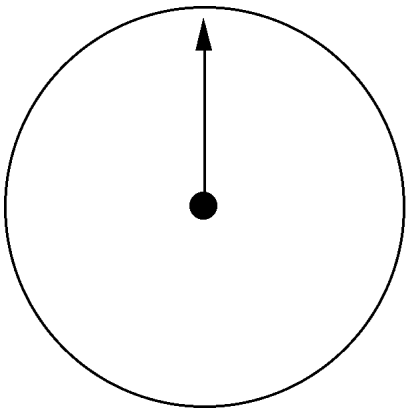
Fig 2.8.1 **The Josephus problem $J_{41}^{(2)}$.**

For the special case of 41 men, with every 2^{nd} man killed (a variation from the historical event), we can readily simulate the entire process. In each cycle of this simulation, the bold numbers are those of the men who are eliminated on that cycle.

1	**2**	3	**4**	$\cdots$	39	**40**	41	0 mod 2
1	3	**5**	7	$\cdots$	**37**	39	**41**	1 mod 4
3	**7**	11	**15**	$\cdots$	**31**	35	**39**	7 mod 8
3	**11**	19	**27**	35				11 mod 16
3	19	**35**						3 mod 32

Thus, the man in position 19 is the survivor.

The survivor position $J_n^{(2)}$ for the first few values of n is given in Figure 2.8.2. Since every man in an even-numbered position is killed on the first cycle, every one of the survivor positions is an odd number.



n	1	2	3	4	5	6	7	8
$J_n^{(2)}$	1	1	3	1	3	5	7	1

Fig 2.8.2 Calculating $J_n^{(2)}$ for small values of n.

After the first traversal of the elimination process around the circle, there are two possible cases, depending on whether the number of men at the outset is odd or

even. If there are $2n$ men at the outset, then after eliminating the even-numbered on the first cycle, the process location immediately precedes position 1. We may regard this as location $2n-1$, with a still-alive occupant, since the occupant of position $2n$ is gone, as shown in Figure 2.8.3. The remaining n men, all odd-numbered, are shown just outside the circle.

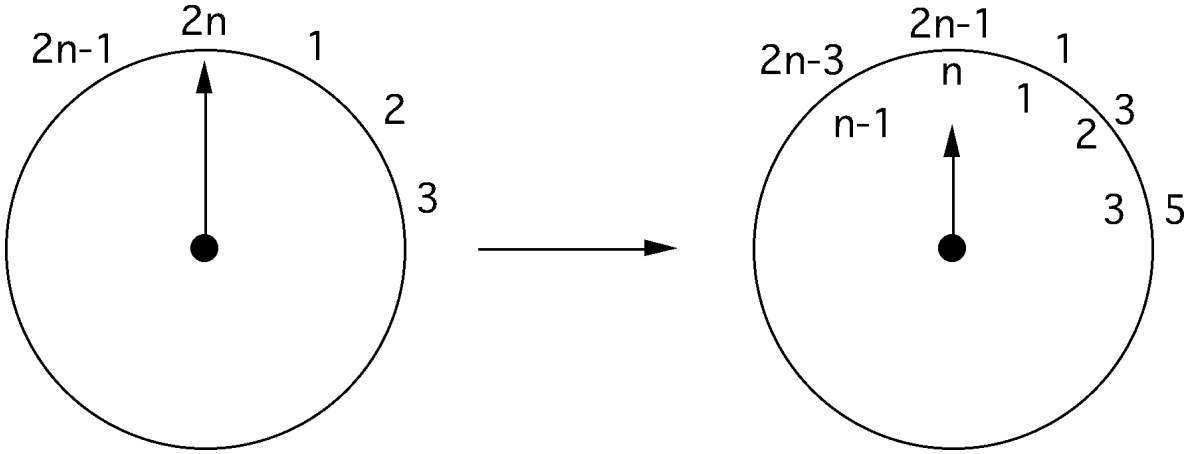


Fig 2.8.3 After one cycle, for an even configuration.

This is equivalent to starting with n men, whose numbers are shown inside the circle. Each outer number is obtained by doubling the inner number and then subtracting 1. Of course, this applies to the survivor position. Thus, we have the recursion

$$J_{2n}^{(2)} = 2J_n^{(2)} - 1 \qquad \text{for } n \geq 1$$

If there are $2n+1$ men at the outset, then after eliminating the even-numbered on the first cycle, the next man to be killed is at position 1. The status of the process

immediately thereafter would be as illustrated in Figure 2.8.4. Here, each outer number is obtained by doubling the inner number and adding 1, which yields the recursion

$$J^{(2)}_{2n+1} = 2J^{(2)}_n + 1 \qquad \text{for } n \geq 1$$

Thus, the recurrence problem to be solved is as follows:

$$\begin{aligned} J^{(2)}_1 &= 1 \\ J^{(2)}_{2n} &= 2J^{(2)}_n - 1 \qquad \text{for } n \geq 1 \\ J^{(2)}_{2n+1} &= 2J^{(2)}_n + 1 \qquad \text{for } n \geq 1 \end{aligned}$$

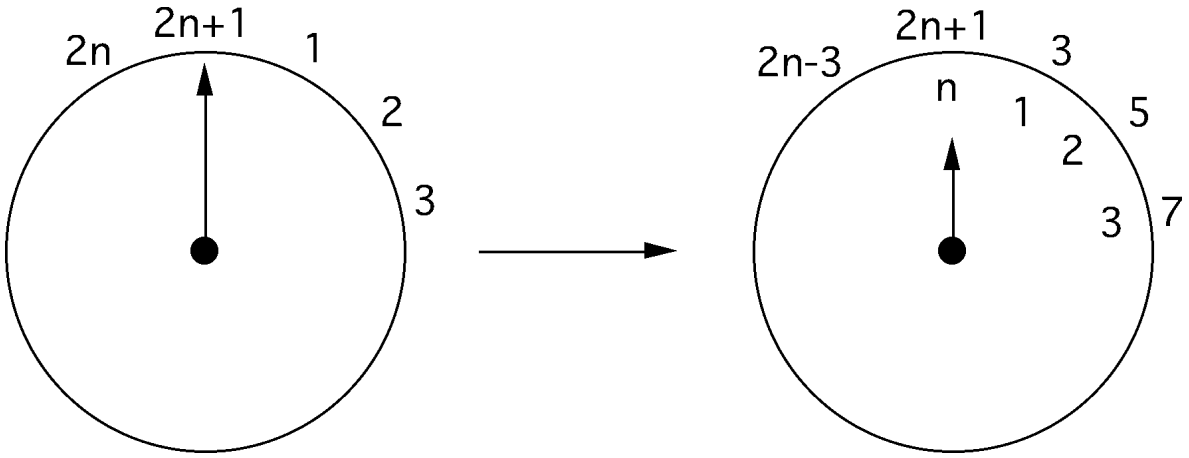


Fig 2.8.4 After one cycle, for an odd configuration.

Applying this divide-and-conquer recursion to $n = 41$ yields a quick solution for that case:

$$\begin{aligned} J_{41}^{(2)} &= 2J_{20}^{(2)} + 1 \\ &= 2(2J_{10}^{(2)} - 1) + 1 \;=\; 4J_{10}^{(2)} - 1 \\ &= 4(2J_{5}^{(2)} - 1) - 1 \;=\; 8J_{5}^{(2)} - 5 \\ &= 8(2J_{2}^{(2)} + 1) - 5 \;=\; 16J_{2}^{(2)} + 3 \\ &= 16(2J_{1}^{(2)} - 1) + 3 \;=\; 32J_{1}^{(2)} - 13 \;=\; 19 \end{aligned}$$

Remark: When the Romans ultimately stormed into the fortress, all the Jews except for Josephus and his friend were dead. Upon hearing from Josephus how he and his friend had survived the suicide pact, the Romans recognized that Josephus was indeed a clever man, who could be quite valuable to them. Josephus lived out his life writing versions of history that flattered the Romans.

To solve the more general problem of calculating $J_n^{(2)}$, we extend the sample of small cases:

n	8	9	10	11	12	13	14	15	16	17	18
$J_n^{(2)}$	1	3	5	7	9	11	13	15	1	3	5

From this increased set of small cases, a pattern emerges, as indicated by the following proposition.

Prop 2.8.1. *If* $n = 2^m + k$, *with* $0 \le k < 2^m$, *then*

$$J_n^{(2)} \;=\; 2k+1 \;=\; 2\left(n \bmod 2^{\lfloor \lg n \rfloor}\right) + 1$$

Proof: By induction on n.

BASIS: The equation is clearly true for $n = 1$.

IND HYP: Assume the equation is true for all cases less than n.

IND STEP: If $n = 2^m + k$ is even, then k is even. Thus,

$$\begin{aligned} J_n^{(2)} &= 2J^{(2)}_{2^{m-1}+\frac{k}{2}} - 1 && \text{(recursion)} \\ &= 2\left(2 \cdot \frac{k}{2} + 1\right) - 1 && \text{(induction hypothesis)} \\ &= 2k+1 \end{aligned}$$

If $n = 2^m + k$ is odd, then $k-1$ is even, and

$$\begin{aligned} J_n^{(2)} &= 2J^{(2)}_{2^{m-1}+\frac{k-1}{2}} + 1 && \text{(recursion)} \\ &= 2\left(2 \cdot \frac{k-1}{2} + 1\right) + 1 && \text{(induction hypothesis)} \\ &= 2k+1 \end{aligned}$$

♢