Section 5.4 Solving Recurrences

Any recursively defined function f with domain **N** that computes numbers is called a *recurrence* or *recurrence relation*. We often denote $f(n)$ by f_n. The goal is to find a closed formula for the general term f_n.

Simple Recurrences

It's easy to solve simple recurrences of the following form, where a_i and b_i are expressions that do not contain r.

$$r_0 = b_0$$
$$r_n = a_n\, r_{n-1} + b_n$$

Substitution Method

Start with the general equation $r_n = a_n\, r_{n-1} + b_n$ and use the definition to keep substituting for r on the right side until we discover a general pattern that allows us to skip ahead and substitute for the basis $r_0 = b_0$.

Example. Solve the recurrence

$$a_0 = 1$$
$$a_n = a_{n-1} + 2n.$$

Solution.
$$\begin{aligned} a_n &= a_{n-1} + 2n \\ &= a_{n-2} + 2(n-1) + 2n \\ &= a_{n-3} + 2(n-2) + 2(n-1) + 2n \\ &\dots \\ &= a_0 + 2 + 2\cdot 2 + \dots + 2(n-2) + 2(n-1) + 2n \\ &= 1 + 2 + 2\cdot 2 + \dots + 2(n-2) + 2(n-1) + 2n \\ &= 1 + 2(1 + 2 + \dots + (n-2) + (n-1) + n) = 1 + n(n+1). \end{aligned}$$

Cancellation Method

Start with the general equation $r_n = a_n\, r_{n-1} + b_n$. Then write a new equation whose left side is the term involving r on the right side of the given equation. Continue this process by writing a new equation whose left side is the term involving r on the right side of the previous equation. Do this until we discover a general pattern that allows us to skip ahead and write the last equation that contains r_0 on the right side. Then add up the equations and cancel the like terms to obtain an expression for r_n.

Example. Solve the recurrence

$$a_0 = 1$$
$$a_n = a_{n-1} + 2n.$$

Solution.

$$\begin{aligned} a_n &= a_{n-1} + 2n \\ a_{n-1} &= a_{n-2} + 2(n-1) \\ a_{n-2} &= a_{n-3} + 2(n-2) \\ &\ldots \\ a_1 &= a_0 + 2(1) \end{aligned}$$

Now add up the equations and cancel the like terms on either side to obtain the following equation for a_n:

$$\begin{aligned} a_n &= a_0 + 2(1) + \ldots + 2(n-2) + 2(n-1) + 2n \\ &= 1 + 2[1 + \ldots + (n-2) + (n-1) + n] \\ &= 1 + n(n+1). \end{aligned}$$

Divide and Conquer Algorithms. Many divide and conquer algorithms are special cases of the following general form. To solve a problem with input of size n split it into s smaller problems each with input of size n/b. To simplify things we assume that $n = b^k$ for two positive integers b and k. Assume that for inputs of size n it takes tn operations to split up the problem and to do other tasks such as assembling the solution. Let a_n be the number of operations to solve the problem with input size n. Then we have the recurrence

$$a_1 = t$$
$$a_n = sa_{n/b} + tn.$$

Solution (by cancellation):

$$\begin{aligned}
a_n &= sa_{n/b} + tn \\
sa_{n/b} &= s^2 a_{n/b^2} + (s/b)tn \\
s^2 a_{n/b^2} &= s^3 a_{n/b^3} + (s/b)^2 tn \\
&\vdots \\
s^{k-1} a_{n/b^{k-1}} &= s^k a_{n/b^k} + (s/b)^{k-1} tn
\end{aligned}$$

Since $n = b^k$, we stop with the last equation.
Now add the equations and cancel to obtain $a_n = s^k a_1 + tn\sum_{i=0}^{k-1}(s/b)^i$.

For example, the closed forms for the cases $s = b$ and $s \neq b$ are as follows:

$$(s = b):\ a_n = b^{\log_b n} a_1 + \sum_{i=0}^{k-1} tn = tn + ktn = tn + (\log_b n)tn = tn(1 + \log_b n).$$

$$(s \neq b):\ a_n = s^k a_1 + tn\sum_{i=0}^{k-1}(s/b)^i = ts^{\log_b n} + tn\left(\frac{(s/b)^{\log_b n} - 1}{(s/b) - 1}\right).$$

Generating Functions

The *generating function* for the infinite sequence $a_0, a_1, \ldots, a_n, \ldots$ is given by the following expression, where x is an indeterminate symbol.

$$\sum_{n=0}^{\infty} a_n x^n.$$

Other names are *formal power series* and infinite polynomial. They can be added, multiplied, divided, and equated just like polynomials.

Some generating functions have closed forms. For example, the generating function for the sequence 1, 1, …, 1, … has the following closed form, which can be verified by multiplying both sides by $1 - x$.

(*Geometric Series Generating Function*) $$\sum_{n=0}^{\infty} x^n = \frac{1}{1-x}.$$

Generating functions with closed forms can be used to solve recurrences.

Example. Suppose someone tells us that the generating function for a_n satisfies the equation

$$\sum_{n=0}^{\infty} a_n x^n = \frac{3}{2x+1}.$$

Notice that $$\sum_{n=0}^{\infty} a_n x^n = \frac{3}{2x+1} = 3\left(\frac{1}{1-(-2x)}\right) = 3\sum_{n=0}^{\infty} (-2x)^n = \sum_{n=0}^{\infty} 3(-2)^n x^n.$$

We can equate coefficients to obtain the solution $a_n = 3(-2)^n$.

Example. Solve the following recurrence.

$a_0 = 3$
$a_1 = 5$
$a_n = 2a_{n-1} + 3a_{n-2}.$

Solution. **Step 1** tells us to use the recurrence to find an algebraic equation in terms of the generating function $A(x)$.

$$A(x) = \sum_{n=0}^{\infty} a_n x^n = a_0 + a_1 x + \sum_{n=2}^{\infty} a_n x^n$$

$$= 3 + 5x + \sum_{n=2}^{\infty} (2a_{n-1} + 3a_{n-2}) x^n$$

$$= 3 + 5x + \sum_{n=2}^{\infty} 2a_{n-1} x^n + \sum_{n=2}^{\infty} 3a_{n-2} x^n$$

$$= 3 + 5x + 2x \sum_{n=2}^{\infty} a_{n-1} x^{n-1} + 3x^2 \sum_{n=2}^{\infty} a_{n-2} x^{n-2}$$

$$= 3 + 5x + 2x \sum_{n=1}^{\infty} a_n x^n + 3x^2 \sum_{n=0}^{\infty} a_n x^n$$

$$= 3 + 5x + 2x(A(x) - a_0) + 3x^2 A(x)$$

$$= 3 + 5x + 2x(A(x) - 3) + 3x^2 A(x).$$

Step 2 tells us to solve the equation for $A(x)$ and try to transform the resulting expression using known generating functions.

$$A(x) = 3 + 5x + 2x(A(x) - 3) + 3x^2 A(x)$$
$$= 3 - x + 2xA(x) + 3x^2 A(x).$$

Collect terms to obtain $A(x)(1 - 2x - 3x^2) = 3 - x.$

Now solve for $A(x)$: $$A(x) = \frac{3-x}{1-2x-3x^2} = \frac{3-x}{(1+x)(1-3x)} = \frac{1}{1+x} + \frac{2}{1-3x}$$
$$= \frac{1}{1-(-x)} + 2 \cdot \frac{1}{1-3x}$$
$$= \sum_{n=0}^{\infty} (-x)^n + 2\sum_{n=0}^{\infty} (3x)^n$$
$$= \sum_{n=0}^{\infty} (-1)^n x^n + 2\sum_{n=0}^{\infty} 3^n x^n$$
$$= \sum_{n=0}^{\infty} ((-1)^n + 2 \cdot 3^n) x^n.$$

Step 3 tells us to equate coefficients to get $a_n = (-1)^n + 2 \cdot 3^n.$

Step 4 tells us to check the answer. (*Quiz*)

Answer. a_0 and a_1 check out OK. Assume $n > 1$ and a_k is OK for $k < n$. Show a_n is OK.

$$a_n = 2a_{n-1} + 3a_{n-2} = 2((-1)^{n-1} + 2 \cdot 3^{n-1}) + 3((-1)^{n-2} + 2 \cdot 3^{n-2}) = ((-1)^n + 2 \cdot 3^n).$$