A summary of recursion solving techniques

Kimmo Eriksson, KTH

January 12, 1999

These notes are meant to be a complement to the material on recursion solving techniques in the textbook *Discrete Mathematics* by Biggs. In particular, Biggs does not explicitly mention the so called Master Theorem, which is much used in the analysis of algorithms. I give some exercises at the end of these notes.

1 Linear homogeneous recursions with constant coefficients

A recursion for a sequence (a_n) of the form

$$a_n = c_{k-1}a_{n-1} + c_{k-2}a_{n-2} + \ldots + c_0 a_{n-k} + f(n)$$

is called a *linear* recursion of order k with constant coefficients. If the term $f(n)$ is zero, the recursion is *homogeneous*.

Linear homogeneous recursions with constant coefficients have a simple explicit general solution in terms of the roots of the *characteristic equation*:

$$x^k - (c_{k-1}x^{k-1} + c_{k-2}x^{k-2} + \ldots + c_0) = 0.$$

Let $P(x)$ be the polynomial in the lefthand part of the equation. Recall that a root r of a polynomial $P(x)$ has *multiplicity* m if the polynomial factors as $P(x) = (x-r)^m Q(x)$ for some polynomial $Q(x)$ and m is the largest such integer.

For example, the polynomial $P(x) = x^3 - x^2 = (x-1)x^2$ has two roots: $r_1 = 1$ of multiplicity 1, and $r_2 = 0$ of multiplicity 2.

Theorem 1 *Let $r_1, \ldots, r_j$ with multiplicities $m_1, \ldots, m_j$ be the roots of the characteristic equation of a linear recursion of order k with constant coefficients. Then the general solution to the recursion is*

$$a_n = P_1(n)r_1^n + P_2(n)r_2^n + \ldots + P_j(n)r_j^n$$

where P_i is an arbitrary polynomial of degree $m_i - 1$ for each $i = 1, \ldots, j$.

1.1 Example

We shall solve the recursion

$$a_n = 4a_{n-1} - 5a_{n-2} + 2a_{n-3}, \quad a_0 = 0, \quad a_1 = 2, \quad a_2 = 3.$$

This is a linear homogeneous recursion of order 3 with constant coefficients. The characteristic equation is

$$P(x) = x^3 - 4x^2 + 5x - 2 = 0.$$

The polynomial $P(x)$ factors as $P(x) = (x-1)^2(x-2)$, so we have roots $r_1 = 1$ of multiplicity $m_1 = 2$ and $r_2 = 2$ of multiplicity $m_2 = 1$. An arbitrary polynomial of degree one $(m_1 - 1)$ is $An + B$. An arbitrary polynomial of degree zero $(m_2 - 1)$ is C. Hence, the theorem gives the general solution

$$a_n = (An + B)1^n + C2^n.$$

The conditions $a_0 = 0$, $a_1 = 2$ and $a_2 = 3$ yield three equations for A, B, C:

$$0 = B + C; \quad 2 = A + B + 2C; \quad 3 = 2A + B + 4C.$$

This is a system of linear equations with the unique solution

$$A = 3, \quad B = 1, \quad C = -1.$$

Therefore the explicit solution to the recursion is

$$a_n = (3n + 1) - 2^n.$$

2 Linear inhomogeneous recursions with constant coefficients

Now suppose that we have a linear *inhomogeneous* recursion with constant coefficients:

$$a_n = c_{k-1}a_{n-1} + c_{k-2}a_{n-2} + \ldots + c_0 a_{n-k} + f(n),$$

where $f(n) \neq 0$. In order to solve such a recursion, we need only solve the corresponding homogeneous recursion and then find *one* particular solution, say a_n^{part}, to the inhomogeneous recursion. Then any solution can be written as

$$a_n = a_n^{\text{hom}} + a_n^{\text{part}},$$

where a_n^{hom} is a solution to the homogeneous recursion. This result follows easily from linearity: If a_n and a_n^{part} both satisfy the inhomogeneous recursion, then subtraction gives

$$a_n - a_n^{\text{part}} = c_{k-1}(a_{n-1} - a_{n-1}^{\text{part}}) + c_{k-2}(a_{n-2} - a_{n-2}^{\text{part}}) + \ldots + c_0(a_{n-k} - a_{n-k}^{\text{part}}).$$

Hence, $a_n^{\text{hom}} := a_n - a_n^{\text{part}}$ satisfies the homogeneous recursion.

So, how does one find a particular solution to an inhomogeneous recursion? Loosely speaking, one tries with some expression of the same form as $f(n)$. However, if such expressions are already solutions to the homogeneous recursion, one must multiply the expression by a polynomial in n.

2.1 Example

We shall find the general solution to the recursion

$$a_n = 4a_{n-1} - 5a_{n-2} + 2a_{n-3} + 3^n.$$

This is a linear inhomogeneous recursion of order 3 with constant coefficients. The inhomogeneous term is $f(n) = 3^n$, so we guess that a particular solution of the form $a_n^{\text{part}} = A \cdot 3^n$ can be found. Plugging this into the recursion gives the equation

$$A \cdot 3^n = 4A \cdot 3^{n-1} - 5A \cdot 3^{n-2} + 2A \cdot 3^{n-3} + 3^n.$$

We simplify by dividing by 3^{n-3}:

$$27A = 36A - 15A + 2A + 27,$$

which has the solution $A = \frac{27}{4}$. Hence a particular solution is $a_n^{\text{part}} = \frac{27}{4}3^n$. The general solution to the corresponding homogeneous recursion was found, in the previous example, to be

$$a_n^{\text{hom}} = An + B + C \cdot 2^n.$$

Hence, the general solution to the inhomogeneous recursion is

$$a_n = a_n^{\text{hom}} + a_n^{\text{part}} = An + B + C \cdot 2^n + \frac{27}{4}3^n.$$

2.2 Example

We shall find the general solution to the recursion

$$a_n = 4a_{n-1} - 5a_{n-2} + 2a_{n-3} + 6.$$

This is a linear inhomogeneous recursion of order 3 with constant coefficients. The inhomogeneous term is $f(n) = 6$, a constant, so we would guess that a constant particular solution could be found. However, $r_1 = 1$ is a root of the characteristic equation so a constant A is already a solution of the homogeneous recursion. Since $r_1 = 1$ has multiplicity 2, also An is a solution of the homogeneous recursion. Hence, we guess a particular solution of the form An^2. Plugging this into the recursion gives the equation

$$An^2 = 4A(n-1)^2 - 5A(n-2)^2 + 2A(n-3)^2 + 7.$$

This equation simplifies to

$$0 = 4A - 20A + 18A + 6,$$

which has the solution $A = -3$. Hence a particular solution is $a_n^{\text{part}} = -3n^2$, so the general solution to the inhomogeneous recursion is

$$a_n = a_n^{\text{hom}} + a_n^{\text{part}} = An + B + C \cdot 2^n - 3n^2.$$

3 The Master Theorem

We now come to a result used in algorithm analysis. When analyzing algorithms that use decomposition, one usually gets recursions of the following form:

$$T(n) = aT(n/b) + F(n), \quad T(1) = d.$$

The term $aT(n/b)$ stands for the time of solving a subproblems of size n/b, to which we add the time $F(n)$ needed to construct the solution to the original problem from the solutions to the subproblems. $T(1) = d$ is the constant time needed to solve a problem of size 1. In computer science, one is interested only in how fast the time $T(n)$ grows and does not care about the explicit expression for $T(n)$. The desired result is called the Master Theorem:

Theorem 2 (Master Theorem) *Suppose that $T(n)$ is given by the recursion $T(n) = aT(n/b) + F(n)$ and $T(1) = d$.*

1. *If $F(n)$ grows* slower *than $n^{\log_b a}$ then $T(n) \in \Theta(n^{\log_b a})$.*

2. *If $F(n) \in \Theta(n^{\log_b a})$ then $T(n) \in \Theta(n^{\log_b a} \log n)$.*

3. *If $F(n)$ grows* faster *than $n^{\log_b a}$, then $T(n) \in \Theta(F(n))$.*

3.1 Example

A version of the *merge sort* algorithm gives the following recursion:

$$T(2n) = 2T(n) + 2n - 1, \quad T(2) = 1.$$

Here we must apply the Master Theorem with parameters $a = 2$, $b = 2$ and $F(n) \in \Theta(n)$. We have $\log_2 2 = 1$, hence we are in case 2. This tells us that

$$T(n) \in \Theta(n \log n).$$

3.2 Sketch of proof

Assume that $n = b^k$, so that $k = \log_b n$. The recursion in the Master Theorem then takes the form

$$T(b^k) = aT(b^{k-1}) + F(b^k).$$

Now make the substitutions $t_k = T(b^k)$ and $f(k) = F(b^k)$. The recursion now takes the familiar form

$$t_k = at_{k-1} + f(k).$$

The solution to the corresponding homogeneous recursion, $t_k = at_{k-1}$, is $t_k^{\text{hom}} = Aa^k$, corresponding to the homogeneous solution $T^{\text{hom}}(n) = Aa^{\log_b n} = An^{\log_b a}$ of the original recursion. We now have three cases depending on how the inhomogeneous term $F(n)$ relates to the homogeneous solution $T^{\text{hom}}(n) = An^{\log_b a}$.

1. If $F(n)$ grows *slower* than $n^{\log_b a}$, then the latter term will dominate, so that $T(n)$ grows as $n^{\log_b a}$.

2. If $F(n)$ grows *equally fast* as $n^{\log_b a}$, then we have asymptotically the situation

   $$T(n) = aT(n/b) + Bn^{\log_b a}, \quad T(1) = d,$$

   which after substitution reads

   $$t_k = at_{k-1} + Ba^k.$$

   This is the case where the homogeneous solution has the same form as the inhomogeneous term, so that the particular solution will be of the form Cka^k. Substituting backwards, this means $C(\log n)n^{\log_b a}$.

3. If $F(n)$ grows *faster* than $n^{\log_b a}$, then $F(n)$ will dominate, so that $T(n)$ will grow as fast as $F(n)$.

4 Other recursions

For other recursions than linear recursions with constant coefficients, explicit solutions may be hard to come by. The method of *generating functions* is always worth trying, though. Briefly, this technique works as follows. Let

$$A(x) = a_0 + a_1 x + a_2 x^2 + \ldots = \sum_{n=0}^{\infty} a_n x^n$$

be the generating function of the sequence $a_0, a_1, a_2, \ldots$ If the recursion can be transformed into an equation for $A(x)$, then we can find the sequence by solving the equation for $A(x)$, and then expanding $A(x)$ into a power series.

4.1 Example

A simple example is the recursion $a_n = a_{n-1}/n$ for $n \geq 1$, and $a_0 = 2$. Multiplying by x^n and summing over n gives

$$A(x) = \sum_{n=0}^{\infty} a_n x^n = a_0 + \sum_{n=1}^{\infty} a_{n-1} \frac{x^n}{n}.$$

Taking the derivative on both sides yields

$$A'(x) = \sum_{n=1}^{\infty} a_{n-1} x^{n-1} = \sum_{n=0}^{\infty} a_n x^n = A(x).$$

Hence, we have obtained the first-order differential equation $A'(x) = A(x)$ with the well-known solution $A(x) = Be^x$. Maclaurin expansion of e^x gives

$$A(x) = B(1/0! + x/1! + x^2/2 + x^3/3! + \ldots) = \sum_{n=0}^{\infty} B \frac{x^n}{n!},$$

and the condition $a_0 = 2$ determines the value of B to be 2. Consequently, we have the solution $a_n = 2/n!$. Of course, this could also have been seen directly from the original recursion.

5 Exercises

1. Show that if two sequences (a_n) and (a'_n) satisfy the same linear recursion, then so does $(Aa_n + A'a'_n)$ for arbitrary constants A and A'.

2. Show that if r is a root to the characteristic equation of a linear recursion with constant coefficients, then the sequence $a_n = r^n$, for $n = 0, 1, 2, \ldots$, satisfies the recursion.

3. The same question for $a_n = n^i r^n$ if r has multiplicity $m > i$.

4. Solve the recursion $a_n = 3a_{n-1} - 2a_{n-2}$, $a_0 = 0$, $a_1 = 1$.

5. Solve the recursion $a_n = 3a_{n-1} + 3^n$, $a_0 = 1$.

6. Solve the recursion $a_n = 2a_{n-1} + 4a_{n-2} - 8a_{n-3} + 1$, $a_0 = a_1 = a_2 = 0$.

7. Show that $a^{\log_b n} = n^{\log_b a}$.

8. A decomposition algorithm for multiplying two integers gives a recursion

$$T(2n) = 3T(n) + 2cn$$

for the time $T(n)$ of multiplying two n-digit integers. (Here c is some constant.) What is the growth rate of $T(n)$?

6 Answers to some of the exercises

1.

2.

3.

4. $a_n = 2^n - 1$.

5. $a_n = (n+1)3^n$.

6. $a_n = (\frac{n}{8} - \frac{5}{16})2^n - \frac{1}{48}(-2)^n + \frac{1}{3}$.

7.

8. $T(n) \in \Theta(n^{\log_2 3})$.