Divide and conquer recurrence

A divide-and-conquer algorithm is a recursive algorithm which

- divides a problem of size n into subproblems, each of size n/b (for simplicity, suppose b divides n)
- and combines the solutions of the subproblems into a solution of the original problem, using $g(n)$ extra operations.

Then, if $f(n)$ counts the number of operations used to solve the problem of size n,

$$f(n) = af(n/b) + g(n).$$

Maximum of a sequence of numbers

Algorithm for locating the maximum of $a_1, a_2, \ldots, a_n$:

If $n = 1$, then the maximum is a_1.

If $n > 1$, split the sequence into two sequences of length $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$ each. Find and compare the maximum of each of the two smaller sequences; output the largest.

If n is even, then the number of operations is given by:

$$f(n) = 2f(n/2) + 1.$$

Merge sort

The merge sort algorithm sorts a list of n elements:

First, split the list into two lists of length $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$ each. Then merge these lists into one sorted list; this uses fewer than n comparisons.

If n is even, then the number of operations is given by:

$$f(n) = 2f(n/2) + n.$$

Fast multiplication of integers

In binary expansion, to multiply two $2n$-bit integers and split each of them into two blocks. The original multiplication is reduced to

three multiplications of n-bit integers,
plus shifts and additions that use Cn operations (for some C).

$$f(n) = 3f(n/2) + Cn.$$

Fast multiplication of matrices

A divide-and-conquer algorithm uses seven multiplications of two $(\frac{n}{2}) \times (\frac{n}{2})$ matrices and 15 additions of $(\frac{n}{2}) \times (\frac{n}{2})$ matrices.

$$f(n) = 7f(n/2) + 15(n/2)^2.$$

Standard multiplication of two $n \times n$ matrix requires $g(n)$ operations where $g(n)$ is $O(n^3)$.

What is the order of $f(n)$?

Asymptotic upper bound of f

Suppose a, c, d are real numbers and b is an integer such that:

$$a \geq 1,\ b > 1,\ c > 0,\ d \geq 0.$$

Master Theorem (Section 8.3)

If $f : \mathbb{N} \to \mathbb{R}$ is an increasing function such that

$$f(n) = af(n/b) + cn^d$$

for all $n = b^k$. Then

$$f(n) \text{ is } \begin{cases} O(n^d) & \text{if } a < b^d \\ O(n^d \log n) & \text{if } a = b^d \\ O(n^{\log_b a}) & \text{if } a > b^d. \end{cases}$$

Applying the master theorem

For fast matrix multiplication:

$$f(n) = 7f(n/2) + 15(n/2)^2.$$

What is the order of $f(n)$?

$$f(n) = 7f(n/2) + 15(n/2)^2$$

The conditions are a, c, d are real numbers and b is an integer such that:

$$a \geq 1,\ b > 1,\ c > 0,\ d \geq 0.$$

$$\begin{aligned} a =& 7 & \geq 1 \\ b =& 2 & > 1 \\ c =& 15/4 & > 0 \\ d =& 2 & \geq 0 \end{aligned}$$

Now compare a and b^d: $7 > 2^2$

Then

$$f(n) \text{ is } O(n^{\log_b a}) = O(n^{2.8}).$$

Asymptotic upper bound of f

Suppose a, c, d are real numbers and b is an integer such that:

$$a \geq 1,\ b > 1,\ c > 0,\ d \geq 0.$$

Master Theorem (Section 8.3)

If $f : \mathbb{N} \to \mathbb{R}$ is an increasing function such that

$$f(n) = af(n/b) + cn^d$$

for all $n = b^k$. Then

$$f(n) \text{ is } \begin{cases} O(n^d) & \text{if } a < b^d \\ O(n^d \log n) & \text{if } a = b^d \\ O(n^{\log_b a}) & \text{if } a > b^d. \end{cases}$$

Solutions for linear homogeneous recurrence

Let c_1 and c_2 be real numbers and consider the equation

$$r^2 - c_1 r - c_2 = 0$$

with roots r_1 and r_2.

Theorem 1 (Section 8.2)

Suppose that r_1 and r_2 are distinct.
Then the sequence $\{a_n\}$ satisfies the recurrence relation

$$a_n = c_1 a_{n-1} + c_2 a_{n-2}$$

if and only if there are constants α_1, α_2 such that

$$a_n = \alpha_1 r_1^n + \alpha_2 r_2^n.$$

Applying the linear recurrence theorem

For Fibonacci numbers:

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

Give an explicit formula for f_n:

Applying the linear recurrence theorem

For $r^2 - 1r - 1 = 0$, the roots are

$$r_1 = \frac{1+\sqrt{1+4}}{2} \text{ and } r_2 = \frac{1-\sqrt{1+4}}{2}.$$

Then

$$f_n = \alpha_1 \left(\frac{1+\sqrt{5}}{2}\right)^n + \alpha_2 \left(\frac{1+\sqrt{5}}{2}\right)^n$$

To find out α_1, α_2, solve the resulting equations when $n = 0, 1$:

Applying the linear recurrence theorem

$$f_0 = 0 = \alpha_1 \left(\frac{1+\sqrt{5}}{2}\right)^0 + \alpha_2 \left(\frac{1-\sqrt{5}}{2}\right)^0 = \alpha_1 + \alpha_2$$

so $\alpha_2 = -\alpha_1$.

$$f_1 = 1 = \alpha_1 \left(\frac{1+\sqrt{5}}{2}\right)^1 - \alpha_1 \left(\frac{1-\sqrt{5}}{2}\right)^1 = \alpha_1\sqrt{5}.$$

so $\alpha_1 = 1/\sqrt{5}$ and $\alpha_2 = -1/\sqrt{5}$. Finally,

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