1 The master method

See section 4.3 of *Introduction to Algorithms* by Cormen, Leiserson, Rivest and Stein for a more extensive exposition.

The master method provides solutions for recurrences of the form

\[ T(n) = aT(n/b) + f(n) \]  

where \( a \geq 1 \) and \( b \geq 1 \) are constants and \( f(n) \) is an asymptotically positive function.

The recurrence (1) describes the running time of an algorithm that divides a problem of size \( n \) into \( a \) subproblems, each of size \( n/b \). The \( a \) subproblems are solved recursively, in time \( T(n/b) \). The cost of dividing the problem and combining the results of the subproblems is given by \( f(n) \).

1.1 The master theorem

The master method is based on the following theorem.

**Theorem 1.1. (Master theorem)**

Let \( a \geq 1 \) and \( b \geq 1 \) be constants, \( f(n) \) a function, and \( T(n) \) defined on the nonnegative integers by:

\[ T(n) = aT(n/b) + f(n) \]

where \( n/b \) means either \( \lfloor n/b \rfloor \) or \( \lceil n/b \rceil \). Then \( T(n) \) can be bounded asymptotically as follows:

1. If \( f(n) = O(n^{\log_b a - \epsilon}) \) for some constant \( \epsilon > 0 \), then \( T(n) = \Theta(n^{\log_b a}) \)
2. If \( f(n) = \Theta(n^{\log_b a}) \), then \( T(n) = \Theta(n^{\log_b a} \log n) \)
3. If \( f(n) = \Omega(n^{\log_b a + \epsilon}) \) for some constant \( \epsilon > 0 \), and if \( af(n/b) \leq cf(n) \) for some constant \( c < 1 \) and all sufficiently large \( n \), then \( T(n) = \Theta(f(n)) \).

Intuitively, the larger function of \( f(n) \) and \( n^{\log_b a} \) determines the solution.
1.2 Proof of the master theorem

We do as in *Introduction to Algorithms*, proving a simpler version of the master theorem for when \( n \) is an exact power of \( b \) in (1).

**Lemma 1.2.** Let \( \alpha \geq 1 \) and \( \beta \geq 1 \) be constants, and \( f(n) \) a nonnegative function defined on exact powers of \( b \). Define \( T(n) \) on exact powers of \( b \) by:

\[
T(n) = \begin{cases} 
\Theta(1) & \text{if } n = 1 \\
\alpha T(n/b) + f(n) & \text{if } n = b^i 
\end{cases}
\]

Then

\[
T(n) = \Theta(n^{\log_b \alpha}) + \sum_{j=0}^{\log_b n-1} \alpha^j f\left(\frac{n}{b^j}\right)
\]  

(2)

**Proof:** We derive the expression for \( T(n) \) by expanding its definition: \( T(n) = \alpha T\left(\frac{n}{b}\right) + f(n) = \alpha (\alpha T\left(\frac{n}{b^2}\right) + f\left(\frac{n}{b}\right)) + f(n) = \alpha^2 T\left(\frac{n}{b^3}\right) + \alpha f\left(\frac{n}{b}\right) + f(n) = \alpha^3 T\left(\frac{n}{b^4}\right) + \alpha^2 f\left(\frac{n}{b^2}\right) + f(n) = \ldots
\)

The expansion continues until \( \frac{n}{b^j} = 1 \), i.e., until \( j = \log_b n \), giving: \( T(n) = \sum_{j=0}^{\log_b n-1} \alpha^j f\left(\frac{n}{b^j}\right) + \alpha^{\log_b n} n T(1) = \sum_{j=0}^{\log_b n-1} \alpha^j f\left(\frac{n}{b^j}\right) + \alpha^{\log_b n} \Theta(1) \).

Finally, since \( \log_b n = \log_b \alpha \cdot \log_\alpha n \), we obtain (2).

Now we give asymptotic bounds for the series \( \sum_{j=0}^{\log_b n-1} \alpha^j f\left(\frac{n}{b^j}\right) \).

**Lemma 1.3.** Let \( \alpha \geq 1 \) and \( \beta \geq 1 \) be constants, and \( f(n) \) a nonnegative function defined on exact powers of \( b \). The function \( g(n) \) defined on exact powers of \( b \) as:

\[
g(n) = \sum_{j=0}^{\log_b n-1} \alpha^j f\left(\frac{n}{b^j}\right)
\]  

(3)

can be bounded asymptotically as follows.

1. If \( f(n) = O(n^{\log_b \alpha - \epsilon}) \) for some constant \( \epsilon > 0 \), then \( g(n) = O(n^{\log_b \alpha}) \)

2. If \( f(n) = \Theta(n^{\log_b \alpha}) \), then \( g(n) = \Theta(n^{\log_b \alpha} \log n) \)

3. If \( \alpha f(n/b) \leq c f(n) \) for some constant \( c < 1 \) and for all \( n \geq b \), then \( g(n) = \Theta(f(n)) \).

**Proof:** For case 1, we substitute \( f(n) = O(n^{\log_b \alpha - \epsilon}) \) into (3), obtaining

\[
g(n) = O\left(\sum_{j=0}^{\log_b n-1} \alpha^j \left(\frac{n}{b^j}\right)^{\log_b \alpha - \epsilon}\right)
\]

Now, \( \sum_{j=0}^{\log_b n-1} \alpha^j \left(\frac{n}{b^j}\right)^{\log_b \alpha - \epsilon} = n^{\log_\alpha \alpha - \epsilon} \sum_{j=0}^{\log_b n-1} \left(\frac{b^j}{\alpha \log \alpha}\right)^j = n^{\log_\alpha \alpha - \epsilon} \sum_{j=0}^{\log_b n-1} (b')^j \),

which we recognize as a geometric series. So we get \( n^{\log_\alpha \alpha - \epsilon} \left(\frac{b'}{\alpha \log \alpha}\right)^{n-1} = n^{\log_\alpha \alpha - \epsilon} \left(\frac{\alpha \log \alpha}{b'} \right)^{n-1} \).

Finally, we get \( n^{\log_\alpha \alpha - \epsilon} O(n') = O(n^{\log_\alpha \alpha}) \).
For case 2, we obtain:

\[
g(n) = \Theta\left(\sum_{j=0}^{\log a \cdot n - 1} a^j \left(\frac{n}{b^j}\right)^{\log_b a}\right)
\]

And \[\sum_{j=0}^{\log a \cdot n - 1} a^j \left(\frac{n}{b^j}\right)^{\log_b a} = n^{\log a} \sum_{j=0}^{\log a \cdot n - 1} \left(\frac{a}{b^j}\right)^j = n^{\log a} \sum_{j=0}^{\log a \cdot n - 1} 1 = n^{\log a} \cdot \log_b a\]

Hence we get \[g(n) = \Theta\left(n^{\log a} \cdot \log_b a\right) = \Theta\left(n^{\log a} \cdot \log \log a\right).\]

For case 3, note that \[f\left(\frac{n}{a}\right) \leq \frac{\log a}{c} f(n)\] implies \[f\left(\frac{n}{a}\right) \leq \frac{\log a}{c} f(n) \leq \left(\frac{\log a}{c}\right)^2 f(n),\] and in general \[f\left(\frac{n}{a}\right) \leq \left(\frac{\log a}{c}\right)^j f(n).\]

Hence, we have: \[g(n) \leq \sum_{j=0}^{\log a \cdot n - 1} c^j f(n) \leq f(n) \sum_{j=0}^{\infty} c^j = f(n) \left(\frac{1}{1-c}\right) = O(f(n)).\]

We also have \[g(n) = \Omega(f(n)),\] since all terms of \[g(n)\] are nonnegative. Thus, \[g(n) = \Theta(f(n)).\]

From the above lemmas, the simpler version of the master theorem is straightforward. Oh, and by the way, the simpler version of the theorem is Lemma 4.4. of Introduction to Algorithms.

1.3 Examples

We illustrate the master method on a couple of examples (from Wikipedia).

1. Consider

\[T(n) = 8T\left(\frac{n}{2}\right) + 100n^2\]

It is an instance of (1) with \[a = 8, b = 2, f(n) = 100n^2, \log_b a = \log_2 8 = 3.\]

The master method says that if \[100n^2 = O(n^{3-\epsilon})\] for some \(\epsilon\), then \[T(n) = \Theta(n^3).\] Can you find such \(\epsilon\)?

2. Consider

\[T(n) = 2T\left(\frac{n}{2}\right) + 10n\]

It is an instance of (1) with \[a = 2, b = 2, f(n) = 10n, \log_b a = \log_2 2 = 1.\]

The master method says that if \[10n = \Theta(n),\] then \[T(n) = \Theta(n \log n).\]

3. Consider

\[T(n) = 2T\left(\frac{n}{2}\right) + n^2\]

is an instance of (1) with \[a = 2, b = 2, f(n) = n^2, \log_b a = \log_2 2 = 1.\]

The master method says that if \[n^2 = \Omega(n^{1+\epsilon})\] for some \(\epsilon\), and if \[2 \left(\frac{1}{2}\right)^2 \leq cn^2\] for some \(c\), then \[T(n) = \Theta(n^2).\] Are there such \(\epsilon\) and \(c\)?
2 Recommended problems

Recommended problems from *Introduction to Algorithms*. I go through some of them…

- On loop invariants and time bounds: exercises 2.2-{1, 2, 3} on p. 27
- On loop invariants: exercise 2.1-3 on p. 21
- On loop invariants and time bounds: problem 2.2 on p. 38
- On recurrence: exercises 2.3-5 on p. 37, 4.3-{1, 3} on p. 75, problem 4-1 on p. 85
- On induction proofs: 2.3-3 on p. 36