

Master Method: An Alternate Formulation

February 5, 2004

Handout 1

Alternate formulation of the Master Method

The formulation presented here will solve recurrences of the form:

$$\begin{aligned} T(n) &= \Theta(1) \text{ for all } n \leq c \\ T(n) &= aT(n/b) + f(n) \text{ for all } n > c \end{aligned}$$

where $f(n) = \Theta(n^\ell(\log n)^k)$ for constants, $c \geq 0$, $a \geq 1$, $b > 1$, $\ell \geq 0$, and $k \geq 0$.

Case 1: For $\ell < \log_b a$, $T(n) = \Theta(n^{\log_b a})$

Case 2: For $\ell = \log_b a$, $T(n) = \Theta(f(n) \log n) = \Theta(n^{\log_b a}(\log n)^{k+1})$

Case 3: For $\ell > \log_b a$, $T(n) = \Theta(f(n)) = \Theta(n^\ell(\log n)^k)$

To get some practice with this, solve the problems in the text book on pages 74 and 75. Check that you get the same answers as given in the text.

Some Things to Remember

For recurrences of the form $T(n) = aT(n/b) + f(n)$, when the base case occurs for inputs of size 1, there are exactly $a^{\log_b n} = n^{\log_b a}$ applications of the base case of the recurrence. So $\Theta(n^{\log_b a})$ time is spent at the base of the recursive calls. In Case 1, the time spent here is as much (asymptotically) as all the time spent dividing and combining.

Just as we saw when analyzing the time complexity of the divide-and-conquer algorithm for finding the closest pair of points, in Case 2, there are $\log_b n$ levels of recursion, across each of which $f(n)$ time is spent splitting and combining.

Finally, in Case 3, the $f(n)$ time spent splitting and combining for the top-level call is as much (asymptotically) as all the rest of the computation.

Reason for Presenting This Formulation

I have selected this formulation to present since students generally find it easier to use than the one in the text book. Also, the version in the text book only considers when $k = 0$. The formulation in the textbook is more general in that it can apply for recurrences of the form $T(n) = aT(n/b) + f(n)$ where $f(n)$ is not of the form $\Theta(n^\ell(\log n)^k)$. However, it is rare to find any divide-and-conquer algorithms where the cost to split and then combine is not of the form $\Theta(n^\ell(\log n)^k)$.