(Particular Cases of the) Master Theorem

Pierre Flener, Ruslan Fomkin, and Justin Pearson

March 19, 2007

Suppose the recursive case of a recurrence is of the form

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$$

meaning that there are $a > 0$ recursive calls, each over an input that is $b > 0$ times smaller than the original input, whereas dividing the input and combining the outputs of the recursive calls take a total of $\Theta(f(n))$ time. The following table gives closed forms for some common cases of this recurrence:

$a = 1$	$b = 2$	$f(n) = \Theta(1)$	$T(n) = \Theta(\log n)$
$a = 1$	$b = 2$	$f(n) = \Theta(n \log n)$	$T(n) = \Theta(n \log n)$
$a = 1$	$b = 2$	$f(n) = \Theta(n^k)$, with $k > 0$	$T(n) = \Theta(n^k)$
$a = 2$	$b = 2$	$f(n) = \Theta(1)$	$T(n) = \Theta(n)$
$a = 2$	$b = 2$	$f(n) = \Theta(\log n)$	$T(n) = \Theta(n)$
$a = 2$	$b = 2$	$f(n) = \Theta(n)$	$T(n) = \Theta(n \log n)$
$a = 2$	$b = 2$	$f(n) = \Theta(n^k)$, with $k > 1$	$T(n) = \Theta(n^k)$

The general case and its mathematics are explained in [1], for instance.

References

[1] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms* (second edition). The MIT Press, 2001. See `http://mitpress.mit.edu/algorithms/`.