Master Theorem

By: Dr. Surjeet Kumar

Dept. of Computer Application

Source: “Introduction to Algorithms” PHI 3rd Edition

by Thomas H. Cormen & Others.

naster theorem

naster method depends on the following theorem.

rem (Master theorem)

≥ 1 and $b > 1$ be constants, let $f(n)$ be a function, and let $T(n)$ be defined e nonnegative integers by the recurrence

$$= aT(n/b) + f(n) ,$$

e we interpret n/b to mean either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Then $T(n)$ has the follow- symptotic bounds:

$f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.

$f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n)$.

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

efore applying the master theorem to some examples, let's spend a moment g to understand what it says. In each of the three cases, we compare the tion $f(n)$ with the function $n^{\log_b a}$. Intuitively, the larger of the two functions rmines the solution to the recurrence. If, as in case 1, the function $n^{\log_b a}$ is the er, then the solution is $T(n) = \Theta(n^{\log_b a})$. If, as in case 3, the function $f(n)$ e larger, then the solution is $T(n) = \Theta(f(n))$. If, as in case 2, the two func- s are the same size, we multiply by a logarithmic factor, and the solution is $) = \Theta(n^{\log_b a} \lg n) = \Theta(f(n) \lg n)$.

eyond this intuition, you need to be aware of some technicalities. In the first , not only must $f(n)$ be smaller than $n^{\log_b a}$, it must be *polynomially* smaller.

t is, $f(n)$ must be asymptotically smaller than $n^{\log_b a}$ by a factor of n^ϵ for some
stant $\epsilon > 0$. In the third case, not only must $f(n)$ be larger than $n^{\log_b a}$, it also
t be polynomially larger and in addition satisfy the "regularity" condition that
$n/b) \le cf(n)$. This condition is satisfied by most of the polynomially bounded
tions that we shall encounter.

ote that the three cases do not cover all the possibilities for $f(n)$. There is
p between cases 1 and 2 when $f(n)$ is smaller than $n^{\log_b a}$ but not polynomi-
smaller. Similarly, there is a gap between cases 2 and 3 when $f(n)$ is larger
$n^{\log_b a}$ but not polynomially larger. If the function $f(n)$ falls into one of these
s, or if the regularity condition in case 3 fails to hold, you cannot use the master
hod to solve the recurrence.

ng the master method

se the master method, we simply determine which case (if any) of the master
rem applies and write down the answer.

s a first example, consider

$$) = 9T(n/3) + n .$$

r this recurrence, we have $a = 9$, $b = 3$, $f(n) = n$, and thus we have that $n^{\log_b a} = n^{\log_3 9} = \Theta(n^2)$. Since $f(n) = O(n^{\log_3 9 - \epsilon})$, where $\epsilon = 1$, we can apply se 1 of the master theorem and conclude that the solution is $T(n) = \Theta(n^2)$.

Now consider

$$T(n) = T(2n/3) + 1,$$

which $a = 1$, $b = 3/2$, $f(n) = 1$, and $n^{\log_b a} = n^{\log_{3/2} 1} = n^0 = 1$. Case 2 plies, since $f(n) = \Theta(n^{\log_b a}) = \Theta(1)$, and thus the solution to the recurrence $T(n) = \Theta(\lg n)$.

For the recurrence

$$T(n) = 3T(n/4) + n \lg n ,$$

have $a = 3$, $b = 4$, $f(n) = n \lg n$, and $n^{\log_b a} = n^{\log_4 3} = O(n^{0.793})$. nce $f(n) = \Omega(n^{\log_4 3 + \epsilon})$, where $\epsilon \approx 0.2$, case 3 applies if we can show that regularity condition holds for $f(n)$. For sufficiently large n, we have that $af(n/b) = 3(n/4)\lg(n/4) \le (3/4)n \lg n = cf(n)$ for $c = 3/4$. Consequently, case 3, the solution to the recurrence is $T(n) = \Theta(n \lg n)$.

The master method does not apply to the recurrence

$$T(n) = 2T(n/2) + n \lg n ,$$

en though it appears to have the proper form: $a = 2$, $b = 2$, $f(n) = n \lg n$, d $n^{\log_b a} = n$. You might mistakenly think that case 3 should apply, since

$(n) = n \lg n$ is asymptotically larger than $n^{\log_b a} = n$. The problem is that it not *polynomially* larger. The ratio $f(n)/n^{\log_b a} = (n \lg n)/n = \lg n$ is asymp-tically less than n^{ϵ} for any positive constant ϵ. Consequently, the recurrence falls to the gap between case 2 and case 3.

Let's use the master method to solve the recurrences

$$T(n) = 2T(n/2) + \Theta(n) ,$$

aracterizes the running times of the divide-and-conquer algorithm for both the aximum-subarray problem and merge sort. (As is our practice, we omit stating e base case in the recurrence.) Here, we have $a = 2$, $b = 2$, $f(n) = \Theta(n)$, and us we have that $n^{\log_b a} = n^{\log_2 2} = n$. Case 2 applies, since $f(n) = \Theta(n)$, and so e have the solution $T(n) = \Theta(n \lg n)$.

$$(n) = 8T(n/2) + \Theta(n^2) ,$$

scribes the running time of the first divide-and-conquer algorithm that we saw r matrix multiplication. Now we have $a = 8$, $b = 2$, and $f(n) = \Theta(n^2)$, d so $n^{\log_b a} = n^{\log_2 8} = n^3$. Since n^3 is polynomially larger than $f(n)$ (that is, $(n) = O(n^{3-\epsilon})$ for $\epsilon = 1$), case 1 applies, and $T(n) = \Theta(n^3)$.

Finally, consider recurrence

$$T(n) = 7T(n/2) + \Theta(n^2) ,$$

which describes the running time of Strassen's algorithm. Here, we have $a = 7$, $b = 2$, $f(n) = \Theta(n^2)$, and thus $n^{\log_b a} = n^{\log_2 7}$. Rewriting $\log_2 7$ as $\lg 7$ and recalling that $2.80 < \lg 7 < 2.81$, we see that $f(n) = O(n^{\lg 7 - \epsilon})$ for $\epsilon = 0.8$. Again, case 1 applies, and we have the solution $T(n) = \Theta(n^{\lg 7})$.

Exercises

Use the master method to give tight asymptotic bounds for the following recurrences.

a. $T(n) = 2T(n/4) + 1$.

b. $T(n) = 2T(n/4) + \sqrt{n}$.

c. $T(n) = 2T(n/4) + n$.

d. $T(n) = 2T(n/4) + n^2$.