1 The Relation of the Two Forms

Description

The master method a.k.a. master theorem is a formula for solving recurrence relations of the form

$$T(N) = aT(N/b) + f(N)$$

where N is the size of input, a is the number of subproblems in the recursion and n/b is the size of each subproblem and $f(N)$ is the cost of the work done outside the recursive call. Let $a \geq 1$ and $b > 1$ be constants, and $f(N)$ is an asymptotically positive funsion, then the time complexity is given by

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

Outline of the Proof

To compute the complexity, we can write every term in the recursion tree form:

$$T(N) = f(N) + af(N/b) + a^2 f(N/b^2) + \ldots + a^L f(N/b^L) + a^L \Theta(1)$$

$\sum_{i=0}^{L} a^i f(N/b^i)$ accounts for the cost in "combination", and $a^L \Theta(1) = \Theta(N^{\log_b a})$ is the cost of "conquering" in all the leaves, which each takes constant time. To prove the master method, we prove that $\sum_0^L a^i f(N/b^i)$ is bounded by some geometric series so it is either a constant times the last term (case 1); $\Theta(Lf(N))$ (case 2); or a constant times the first term (case 3). See our textbook for the proofs.

Regularity Condition

In case 3, $af(N/b) < cf(N)$ for some $c < 1$ is called the regularity condition, which itself guarantees $f(N) = O(N^{\varepsilon + \log_b a})$:

$$f(N) > \frac{a}{c} f(N/b) > (\frac{a}{c})^L f(N/b^L)$$

$L = \log_b N$ and since the number of elements on the leaves are small enough, $f(N/b^L) = \Theta(1)$

$$(\frac{a}{c})^L f(N/b^L) = (\frac{a}{c})^{\log_b N} \Theta(1) = \Theta(N^{\log_b a + \log_b \frac{1}{c}}) = \Theta(N^{\log_b a + \varepsilon})$$

The last equality comes from $0 < c < 1$. Thus $f(N) = \Omega(N^{\log_b a+\varepsilon})$.

Second Form Rewritten

Another form can be described as follows:

1. If $af(N/b) \geq Kf(N)$ for some constant $K > 1$, then $T(N) = \Theta(N^{\log_b a})$.
2. If $af(N/b) = f(N)$, then $T(N) = \Theta(f(N) \log_b N)$.
3. If $af(N/b) \leq \kappa f(N)$ for some constant $\kappa < 1$, then $T(N) = \Theta(f(N))$.

It's different from the slides that I rearranged the cases and changed $=$ into $\leq$ and $\geq$. In the two forms, case 3 are the same. In addition, case 1 and case 2 of the second form seems to be stronger than the first form. That is, $af(N/b) \geq Kf(N)$ leads to $f(N) = O(N^{-\varepsilon+\log_b a})$ and $af(N/b) = f(N)$ leads to $f(N) = \Theta(N^{\log_b a})$. What about the other way around?

2 Proof of the Log Case

Especially, when $f(N) = \Theta(N^k \log^p N)$ and $a = b^k$,

$$T(N) = \begin{cases} \Theta(N^k \log^{p+1} N) & \text{if } p > -1 \\ \Theta(N^k \log\log N) & \text{if } p = -1 \\ \Theta(N^k) & \text{if } p < -1 \end{cases}$$

There is a proof online which I find questionable. It seems to rely on finding a constant D_2 small enough so that $\frac{1}{D_2} > \sum_{i=1}^{k} i \log b$ for all arbitrarily large N, which does not seem possible.

For the case $p > -1$, it's easy to prove $f(N) = O(f(N) \log N)$ but can be challenging to prove $f(N) = \Omega(f(N) \log N)$. The following is one feasible way by substitution method:

Suppose $\exists D > 0$, $\forall m < N$, $T(m) \geq Dm^k \log^{p+1} m$, then

$$\begin{aligned} T(N) &= f(N) + aT(N/b) \\ &\geq N^k \log^p N + aD(\frac{N}{b})^k \log^{p+1} \frac{N}{b} \\ &=\geq N^k \log^p N + DN^k \log^{p+1} N(1 - \frac{\log b}{\log N})^{p+1} \\ &\geq N^k \log^p N + DN^k \log^{p+1} N(1 - (p+1)\frac{\log b}{\log N}) \\ &= DN^k \log^{p+1} N + N^k \log^p N(1 - D(p+1)\log b) \\ &\geq DN^k \log^{p+1} N \end{aligned}$$

if $D < \frac{1}{(p+1)\log b}$.