Analysis of Algorithms

Master's Theorem

Master’s Theorem

- Master’s theorem is one of the many methods that are applied to calculate time complexities of algorithms.
- In analysis, time complexities are calculated to find out the best optimal logic of an algorithm.
- Master’s theorem is applied on recurrence relations.

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

$f(n) = \Theta(n^k \log^p n)$

Example: $T(n) = 8T(n/2) + n^2 \log n$

where n = size of the problem

a = number of subproblems in the recursion and a >= 1

n/b = size of each subproblem

b > 1, k >= 0 and p is a real number.

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

1. if $a > b^k$, then $T(n) = \theta(n^{\log_b a})$

2. if $a = b^k$, then
   (a) if $p > -1$, then $T(n) = \theta(n^{\log_b a} \log^{p+1} n)$
   (b) if $p = -1$, then $T(n) = \theta(n^{\log_b a} \log\log n)$
   (c) if $p < -1$, then $T(n) = \theta(n^{\log_b a})$

3. if $a < b^k$, then
   (a) if $p >= 0$, then $T(n) = \theta(n^k \log^p n)$
   (b) if $p < 0$, then $T(n) = \theta(n^k)$

Example 1

Consider a recurrence relation given as $T(n) = 8T(n/2) + n^2$

In this problem, a = 8, b = 2 and $f(n) = \Theta(n^k \log_n p)$ = n2, giving us k = 2 and p = 0.

$a = 8 > b^k = 2^2 = 4$,

case 1 must be applied for this equation

To calculate, $T(n) = \Theta\ (n\ log_b a)$

$= n^\wedge\ log_2 8$

$= n^\wedge log_2 2^3$

= n^3

Therefore, T(n) = Θ(n^3) is the tight bound for this equation.

Example 2

T(n)=16 T(n/4)+n

a=16 ; b=4 ; k=1 ; p=0

$a>b^k$

16>4

Case 1: $\mathrm{Tn} = \theta(n\ log_b a)$

$= (n\ log_4 16)$

$= (n\ log_4 4^2)$

$= n^2$

$= (n^2)$

Example 3

$T(n) = 3T(n/2) + n^2$

$a = 3,\ b = 2,\ k = 2\ \&\ p = 0$

and $a < b^k \Rightarrow 3 < 2^2$

Case 3:

$$\text{so, } T(n) = \theta(n^k log^p n)$$

$$= \theta(n^2 log^0 n)$$

$$= \theta(n^2)$$