Master Theorem

- Let $a>=1$ and $b>=1$ be two constant, let $f(n)$ be a function and let $T(n)$ be defined on the non negative integers by the recurrences

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

- Where we interpret (n/b) as Floor(n/b) or ceiling(n/b) then T(n) can be bounded asymptotically as follows.

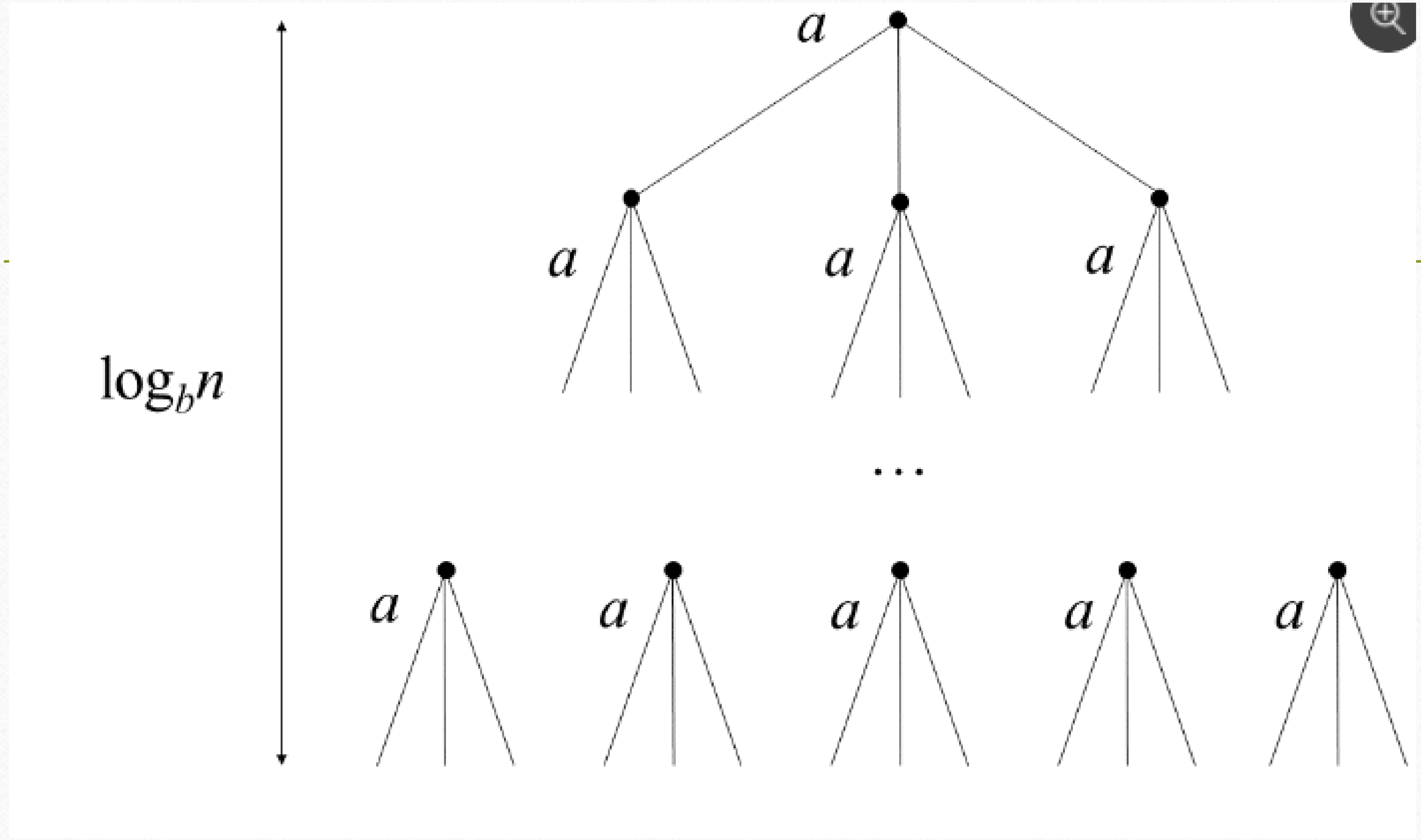
Statement of the Master Theorem

- First, consider an algorithm with a recurrence of the form

$$T(n) = aT\left(\frac{n}{b}\right)$$

- where **a** represents the number of children each node has, and the runtime of each of the three initial nodes is the runtime of $T\left(\frac{n}{b}\right)$.

The tree has a depth of $\log_b n$ and depth i contains a^i nodes. So there are $a^{\log_b n} = n^{\log_b a}$ leaves, and hence the runtime is $\Theta\left(n^{\log_b a}\right)$



$\log_b n$
a
a
a
a
$\cdots$
a
a
a
a
a

Rule 1

Case 1. If $f(n) = O\left(n^{\log_b a - \epsilon}\right)$ for some $\epsilon > 0$, then $T(n) = \Theta\left(n^{\log_b a}\right)$.

Rule 2

Case 2. If $f(n) = \Theta\left(n^{\log_b a}\right)$, then $T(n) = \Theta\left(n^{\log_b a} \log n\right)$.

Rule 3

Case 3. If $f(n) = \Omega\left(n^{\log_b a+\epsilon}\right)$ for some $\epsilon > 0$ $\left(\text{and } af\left(\frac{n}{b}\right) \leq cf(n) \text{ for some } c < 1 \text{ for all } n \text{ sufficiently large}\right)$,

then $T(n) = \Theta\left(f(n)\right)$

Exmp-1:

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

Solⁿ: $T(n) = 4T(n/2) + n.$

Compare it with, $T(n) = aT(n/b) + f(n).$

we get $a = 4$

$b = 2$

$f(n) = n.$

Is $n = O(n^{\log_2 4 - \epsilon}) = O(n^{\log_2 2^2 - \epsilon})$

$= O(n^{2\log_2 2 - \epsilon})$

$= O(n^{2-\epsilon})$

$n = O(n^{2-\epsilon})$

Yes, so Case 1 applies and thus from case 1.

$f(n) = O(n^{\log_b a - \epsilon})$ | $T(n) = \Theta(n^{\log_2 4})$

Example-2:

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

Solⁿ:- Given recurrence is.

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

Compare it with $\boxed{T(n) = aT(n/b) + f(n)}$

Now, we get $a = 4$

$b = 2$

$f(n) = n^2$.

$$n^2 = O(n^{\log_b^a - \epsilon}) = O(n^{\log_2^4 - \epsilon})$$

$$= O(n^{\log_2 2^2 - \epsilon}) = O(n^{2\log_2^2 - \epsilon})$$

$$= O(n^{2.1 - \epsilon}) = O(n^{2-\epsilon})?$$

NO, if $\epsilon > 0$, but it is true only if $\epsilon = 0$.

This shows that case 2 of master theorem is applied.

$$f(n) = n^2$$

$$\Theta(n^{\log_b^a})$$

$$= \Theta(n^{\log_2^4})$$

$$= \Theta(n^{\log_2 2^2}) = \Theta(n^{2\log_2^2})$$

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

Hence,

$$T(n) = \Theta(n^{\log_b^a} \log n)$$

$$\Rightarrow T(n) = \Theta(n^2 . \log n).$$

$$\Rightarrow \boxed{T(n) = \Theta(n^2 \log n}$$

Q Given that $T(n) = 2T(n/2) + n^3$ ——— ①

Soln: Compare it with $T(n) = aT(n/b) + f(n)$.

we have, $a=2$, $b=2$, $f(n) = n^3$

Now, we apply cases for master theorem.

$n^{\log_b a} = n^{\log_2 2} = n^1 = n$.

This satisfies case 3 of master theorem.

$\Rightarrow f(n) = \Omega(n^{\log_b a + \epsilon})$

$= \Omega(n^{1+\epsilon})$

$= \Omega(n^{1+2})$

$= \Omega(n^3)$

Again $2f\left(\frac{n^3}{2}\right) < c \cdot f(n^3)$ ——— ②

eqn ② is true for $c = 2$.

$\Rightarrow T(n) = \Theta(f(n))$

$\Rightarrow \boxed{T(n) = \Theta(n^3)}$

(b) $T(n) = T\left(\frac{9n}{10}\right) + n$.

Compare it with $T(n) = aT(n/b) + f(n)$

we get $a = 1$

$b = \frac{10}{9}$ and $f(n) = n$.

Now, $n^{\log_b a} = n^{\log_{10/9} 1} = n^0 = 1$.

$\Rightarrow f(n) = \Omega(n^{\log_b a + \epsilon})$

$\Rightarrow f(n) = \cancel{g} \cdot \Omega(n^{0+\epsilon}) = \Omega(n^{0+1}) = \Omega(n)$

here, $\epsilon = 1$.

Is $af(n/b) \leq cf(n)$?

$1 f\left(\frac{9n}{10}\right) \leq c \cdot f(n)$.

$\Rightarrow \frac{9n}{10} = c \cdot n$. $\Rightarrow c = \frac{9n}{10n} = \frac{9}{10}$.

$\Theta(f(n)) \Rightarrow \boxed{T(n) = \Theta(n)}$

$$T(n) = 16T(n/4) + n^2$$

$$T(n) = 7T\left(\frac{n}{3}\right) + n^2$$

$$T(n) = 7T(n/2) + n^2$$

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

$$T(n) = T(n-1) + n$$ (apply iteration method)

$$T(n) = T(\sqrt{n}) + 1.$$