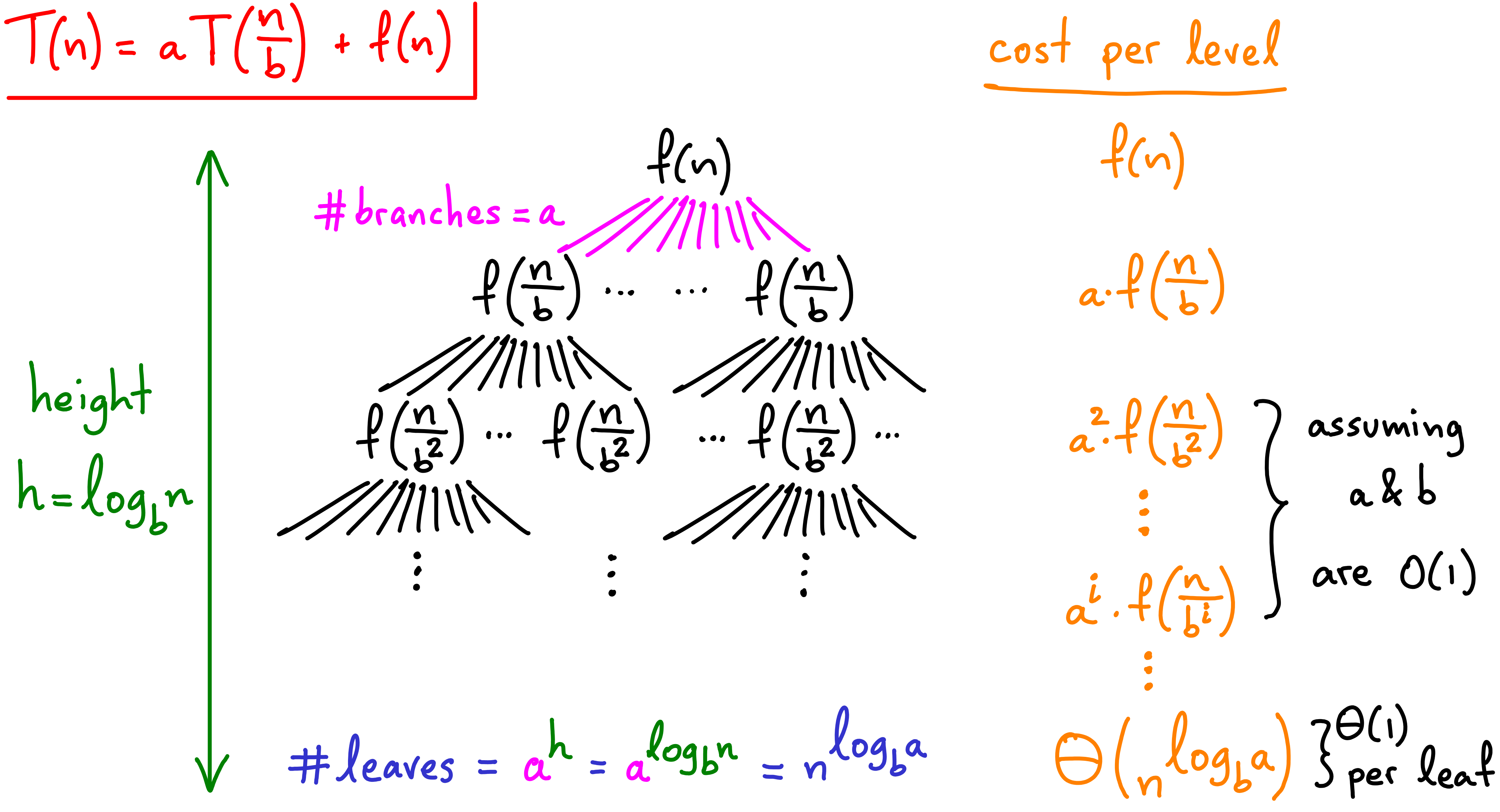
# MASTER METHOD

↳ a tool for solving recurrences of this form:

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

required $\left\{ \begin{array}{lll} a \geq 1 & \longrightarrow & \text{must recurse at least once.} \\ b > 1 & \longrightarrow & \text{otherwise } T(n) = \infty \\ a \text{ \& } b \text{ are } O(1) & \longrightarrow & \text{see next page} \end{array} \right.$

also $f(n) > 0$ for $n > n_0$



$T(n) = aT\left(\frac{n}{b}\right) + f(n)$
cost per level
$f(n)$
# branches = a
$f\left(\frac{n}{b}\right)$ ... ... $f\left(\frac{n}{b}\right)$
$f\left(\frac{n}{b^2}\right)$ ... $f\left(\frac{n}{b^2}\right)$ ... $f\left(\frac{n}{b^2}\right)$ ...
height
$h = \log_b n$
$a \cdot f\left(\frac{n}{b}\right)$
$a^2 \cdot f\left(\frac{n}{b^2}\right)$
$a^i \cdot f\left(\frac{n}{b^i}\right)$
assuming a & b are O(1)
$\#\text{leaves} = a^h = a^{\log_b n} = n^{\log_b a}$
$\Theta\left(n^{\log_b a}\right)$
Θ(1) per leaf

# MASTER METHOD $T(n) = aT\left(\frac{n}{b}\right) + f(n)$

compare $f(n)$ to $n^{\log_b a}$

root #leaves

---

1) $n^{\log_b a} = \Omega(f(n) \cdot n^{\varepsilon})$ $(\varepsilon > 0)$ #leaves dominate polynomially

e.g.: $\#leaves = n^2$, $f(n) = 30n^{1.5} \cdot \log^2 n$ solution: $T(n) = \Theta(n^{\log_b a})$

---

2) $f(n) = \Theta(n^{\log_b a})$ all levels ~ same

e.g.: $\#leaves = n^3$, $f(n) = 2n^3$ solution: $T(n) = \Theta(f(n) \cdot \log n)$

---

3) $f(n) = \Omega(n^{\log_b a} \cdot n^{\varepsilon})$ $(\varepsilon > 0)$ root dominates polynomially

e.g.: $\#leaves = n^4$, $f(n) = n^5$ solution: $T(n) = \Theta(f(n))$

# Technicality for case 3

Also required: $af\left(\frac{n}{b}\right) \leqslant \delta \cdot f(n)$ $\qquad 0 < \delta < 1$

Good news: For commonly encountered functions this will hold.

You don't need to check this condition.

# Divide & Conquer

$$T(n) = 2T\left(\frac{n}{2}\right) + f(n)$$

$$n^{\log_b a} = n$$

$$T(n) = 4T\left(\frac{n}{4}\right) + f(n)$$

SAME

if $f(n) = \Theta(n)$ $\rightarrow$ case 2 $\Theta(n \log n)$

if $f(n) = O(n^d)$ $(d<1)$ $\rightarrow$ case 1 $\Theta(n)$

e.g., $O(1)$, $O(\log n)$, $O(\sqrt{n})$

if $f(n) = \Omega(n^d)$ $(d>1)$ $\rightarrow$ case 3 $\Theta(f(n))$

$$n^{\log_b a} = n^{\log_2 4} = n^2$$

$$T(n) = 4T\left(\frac{n}{2}\right) + n \qquad = \Theta(n^2)$$

leaves dominate polynomially

$$T(n) = 4T\left(\frac{n}{2}\right) + n^2 \qquad = \Theta(n^2 \log n)$$

case 2

$$T(n) = 4T\left(\frac{n}{2}\right) + n^3 \qquad = \Theta(n^3)$$

root dominates polynomially

# EXTENDED CASE 2

$f(n) = \Theta(n^{\log_b a} \cdot \log^k n)$ $\quad k \geq 0 \quad$ ($k=0$ is regular case 2)

$\hookrightarrow T(n) = \Theta(f(n) \cdot \log n)$ $\quad$ same result as regular case 2

Examples:

$T(n) = 2T(\frac{n}{2}) + n\log^5 n \quad = \Theta(n\log^6 n)$

$T(n) = 4T(\frac{n}{2}) + n^2\log n \quad = \Theta(n^2\log^2 n)$

$T(n) = T(\frac{n}{6}) + \log^2 n \quad = \Theta(\log^3 n)$

$T(n) = 2T(\frac{n}{2}) + \frac{n}{\log n}$ $\quad$ N/A $\quad$ *based on what we've seen

FYI - EXTRA-EXTENDED CASE 2 $f(n) = \Theta(n^{\log_b a} \cdot \log^k n)$

(Doesn't come up in any algorithms that we will see)

Standard extended case 2

$k \geq 0$ $T(n) = \Theta(n^{\log_b a} \cdot \log^{k+1} n)$ $T(n) = \Theta(f(n) \cdot \log n)$

$k = -1$ $T(n) = \Theta(n^{\log_b a} \cdot \log\log n)$ $T(n) = \Theta(f(n) \cdot \log n \cdot \log\log n)$

e.g., $T(n) = 8T(\frac{n}{2}) + \frac{n^3}{\log n}$ $= n^3 \log\log n$

$k \leq -2$ $T(n) = \Theta(n^{\log_b a})$ almost like an extended case 1:

Leaf level dominates by a "large" poly-log factor.