S.KIMMEL

Master Method

Way to solve certain recurrences

$$T(n) = a\,T\left(\frac{n}{b}\right) + O(n^d)$$
$$T(n) \le C \quad \text{for } n < n^*$$

a, b, d don't depend on n

Q: If $T(n)$ is runtime of an algorithm,

what are a, b, d in words?

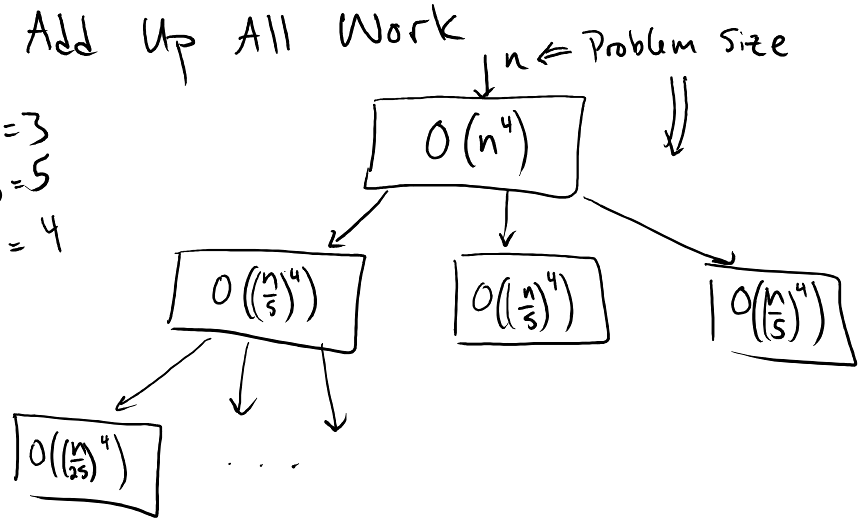
A: a: # of recursive calls

b: factor by which problem shrinks in recursive call

d: characterizes extra work outside recursive call

Let's Add Up All Work

ex: $a=3$
$b=5$
$d=4$



Input size

n

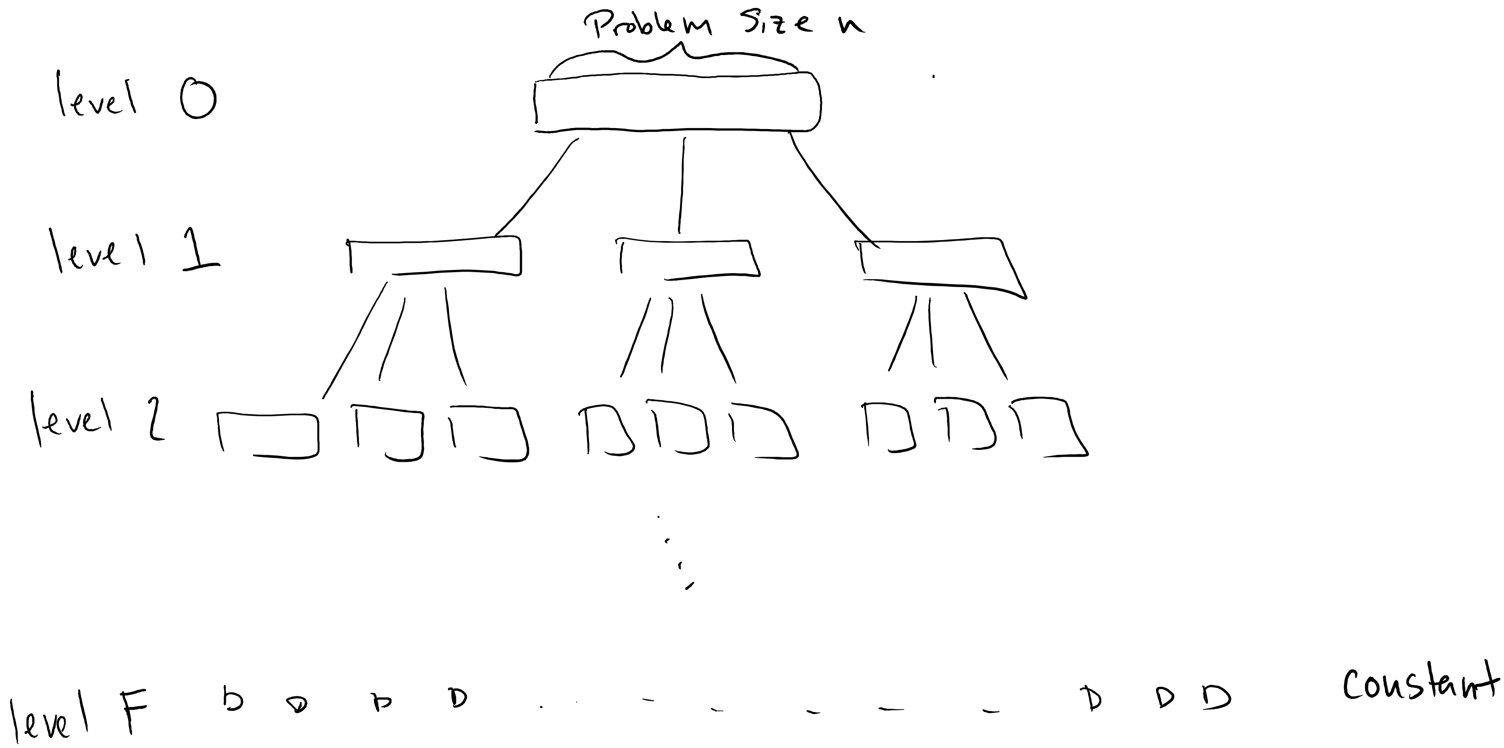
$\frac{n}{5}$

$\frac{n}{25}$

⋮

S.KIMMEL

Proof of Master Method



Q. What is F (in terms of a, b, d)?

A) $O(\log_b n)$ B) $O(\log_d n)$ C) $O(n^{\log_b d})$ D) $O(b^{\log_d n})$

Because at each level, problem size is divided by b. $\log_b n$ is number of times n can be divided by b before reaching a constant.

$$\text{constant} \cdot \underbrace{b \cdot b \cdots b}_{F} = n$$

$$c\,b^F = n$$

$$b^F = \frac{n}{c} \quad \text{take } \log_b \text{ of both sides} \rightarrow F \overset{1}{\log_b b} = \log_b n - \log_b c$$

$$F = \log_b n - \log_b c \qquad F = \log_b n + \text{constant}$$

S.KIMMEL

Q. What is the ~~total~~ work (operations) done at level K (outside of recursive calls & in terms of a, b, d)?

- $\boxed{a^K}$ subproblems at level K.
- level K subproblem size: $\boxed{\frac{n}{b^K}}$
- Work outside of recursive call required to solve 1 subproblem : $\boxed{\left(\frac{n}{b^K}\right)^d}$

$\Rightarrow$ Total work $a^K\left(\frac{n}{b^K}\right)^d = \boxed{\left(\frac{a}{b^d}\right)^K n^d}$

Now we add up work done at all levels:

$$\sum_{K=0}^{\log_b n}\left(\frac{a}{b^d}\right)^K n^d$$

$$T(n) = n^d\left[\sum_{K=0}^{\log_b d}\left(\frac{a}{b^d}\right)^K\right]$$

Mutliplicative Distributive property

Geometric Series:

$$\sum_{k=0}^{F} r^k = \begin{cases} F+1 & \text{if } r = 1 \\ \dfrac{1-r^{F+1}}{1-r} & \text{otherwise} \end{cases}$$

S.KIMMEL

2 cases:

- $\frac{a}{b^d} = 1 \longrightarrow n^d \sum_{k=0}^{\log_b n} \left(\frac{a}{b^d}\right)^k = O\left(n^d \log_b n\right)$

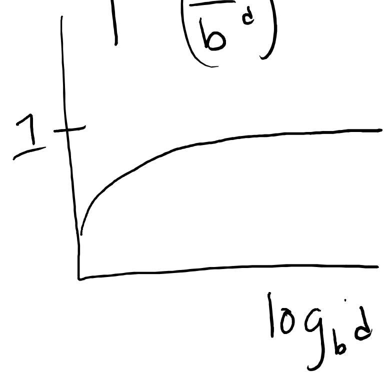
- $\frac{a}{b^d} \neq 1 \longrightarrow n^d \sum_{k=0}^{\log_b n} \left(\frac{a}{b^d}\right)^k = n^d \left(\frac{1-\left(\frac{a}{b^d}\right)^{\log_b n}}{\boxed{1-\frac{a}{b^d}}}\right)$

$\Uparrow$ constant $= C$

Look at:

2 cases

- $\left(\frac{a}{b^d}\right) < 1 \longrightarrow$ $1-\left(\frac{a}{b^d}\right)^{\log_b n} = O(1)$



$T(n) = O(n^d)$

- $\left(\frac{a}{b^d}\right) > 1$ $\quad \frac{1-\left(\frac{a}{b^d}\right)^{\log_b n}}{1-\frac{a}{b^d}} = O\left(\left(\frac{a}{b^d}\right)^{\log_b n}\right)$

S. KIMMEL

$\left(\frac{X}{Z}\right)^{y} = X^{y}/Z^{y}$

$x^{yz} = x^{zy}$

$$\left(\frac{a}{b^d}\right)^{\log_b n} = \frac{a^{\log_b n}}{b^{d \log_b d}} = \frac{a^{\log_b n}}{\left(b^{(\log_b n)}\right)^d}$$

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

$x^{\log_y z} = z^{\log_y x}$

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

$$T(n) = O\left(n^d\left(\frac{n^{\log_b a}}{n^d}\right)\right) = O\left(n^{\log_b a}\right)$$

$$T(n) = aT\left(\frac{n}{b}\right) + O(n^d)$$

$$T(n) = \begin{cases} a = b^d & O(n^d \log n) \\ a < b^d & O(n^d) \\ a > b^d & O(n^{\log_b a}) \end{cases}$$

$O(n^{\log_b a})$ = $a^{\log_b n}$ = # of vertices in final level of tree

Q: Interpret

- Balance between current work + recursive work.
- Run time dominated by work outside recursive calls
- Runtime dominated by work in bottom level of tree