S.KIMMEL

Tree Method

Way to solve certain recurrences

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

a, b, d don't depend on n

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

what are a, b, d in words?

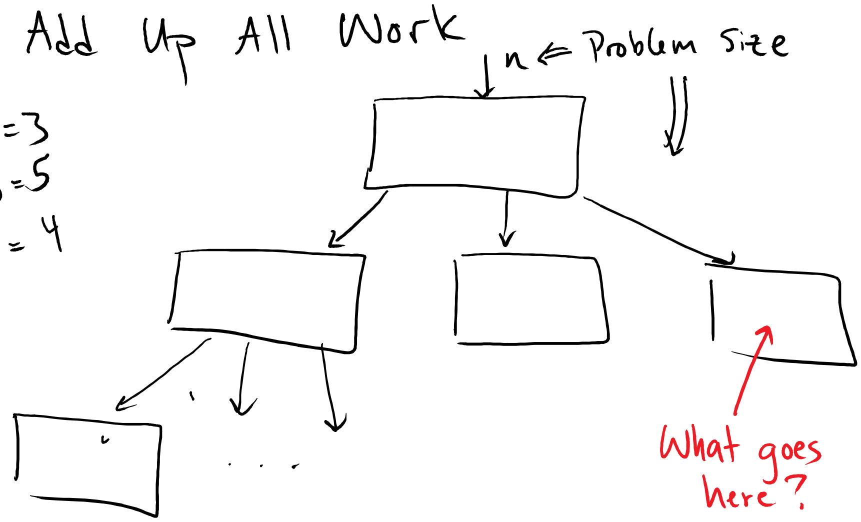
A: a: # of recursive calls

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

$\frac{n}{5}$

$\frac{n}{25}$

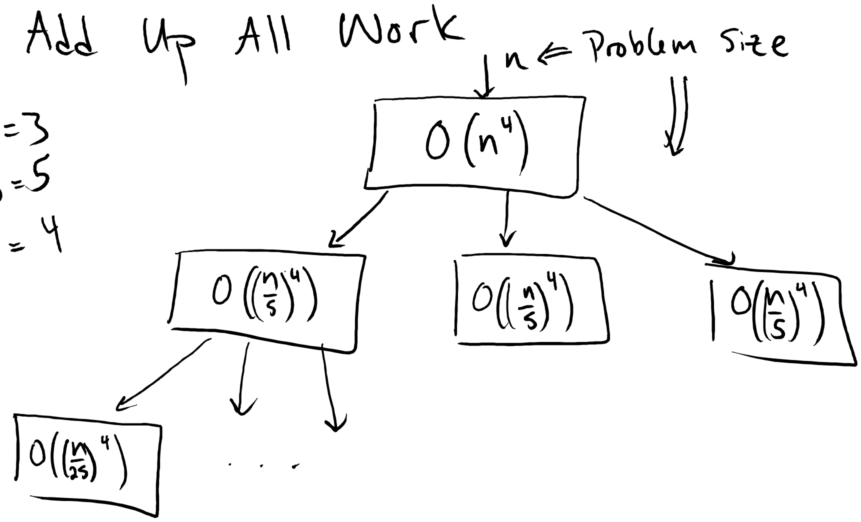
⋮

A) $O\left(\frac{n}{5}\right)$ B) $O\left(\frac{n}{2}\right)$ C) $O\left(\left(\frac{n}{5}\right)^4\right)$ D) $O\left(\left(\frac{n}{3}\right)^4\right)$

S. KIMMEL

Let's Add Up All Work

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



Input size

n

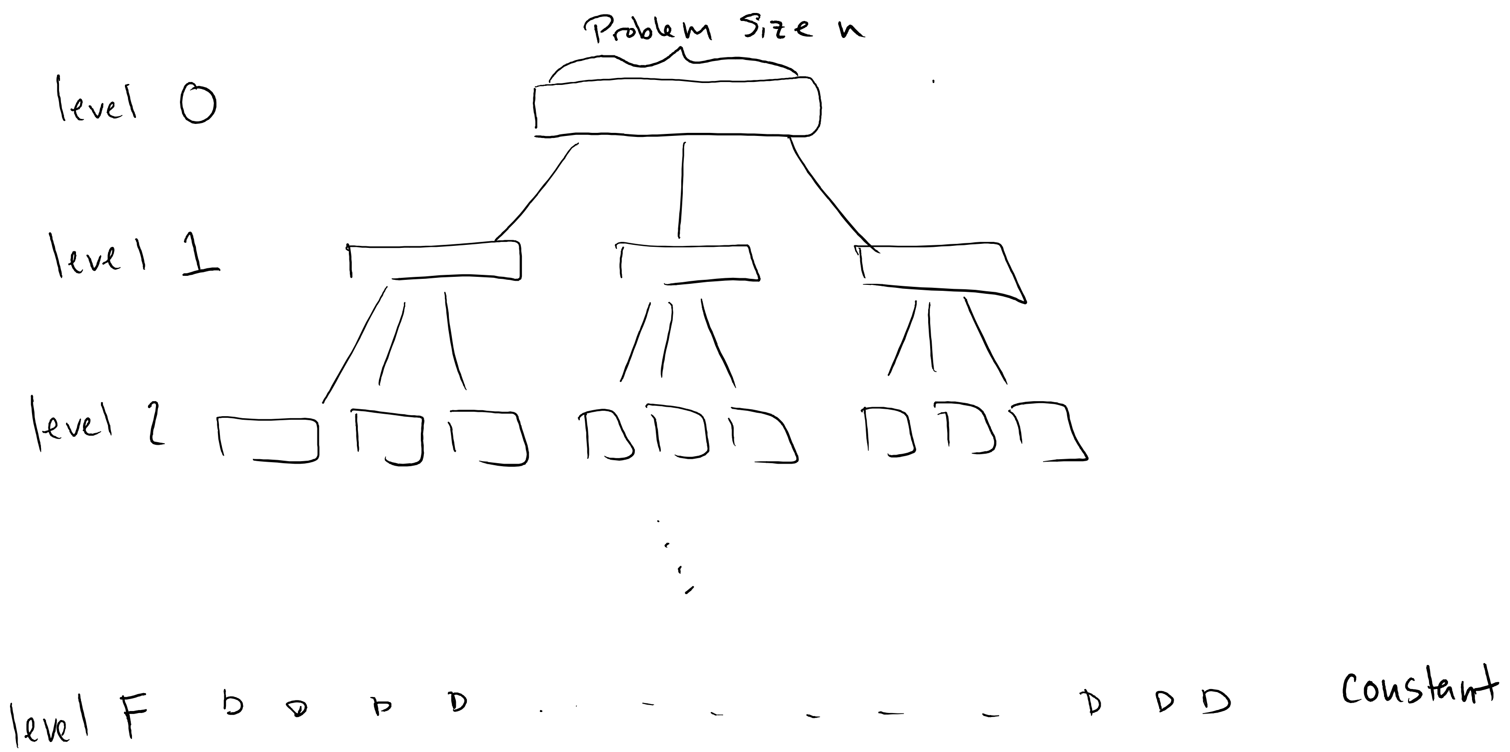
$\frac{n}{5}$

$\frac{n}{25}$

$\vdots$

S.KIMMEL

Proof of Tree 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.

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

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

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

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

S.KIMMEL

Q. What is the total work $\overset{\swarrow \text{ operations}}{}$ done just at level K, not at other levels?

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

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

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

PSET:

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

↑

This is usually called "master method"
"master theorem"

Master has pretty unpleasant connotations. Also it is not descriptive

My term: "Tree method"