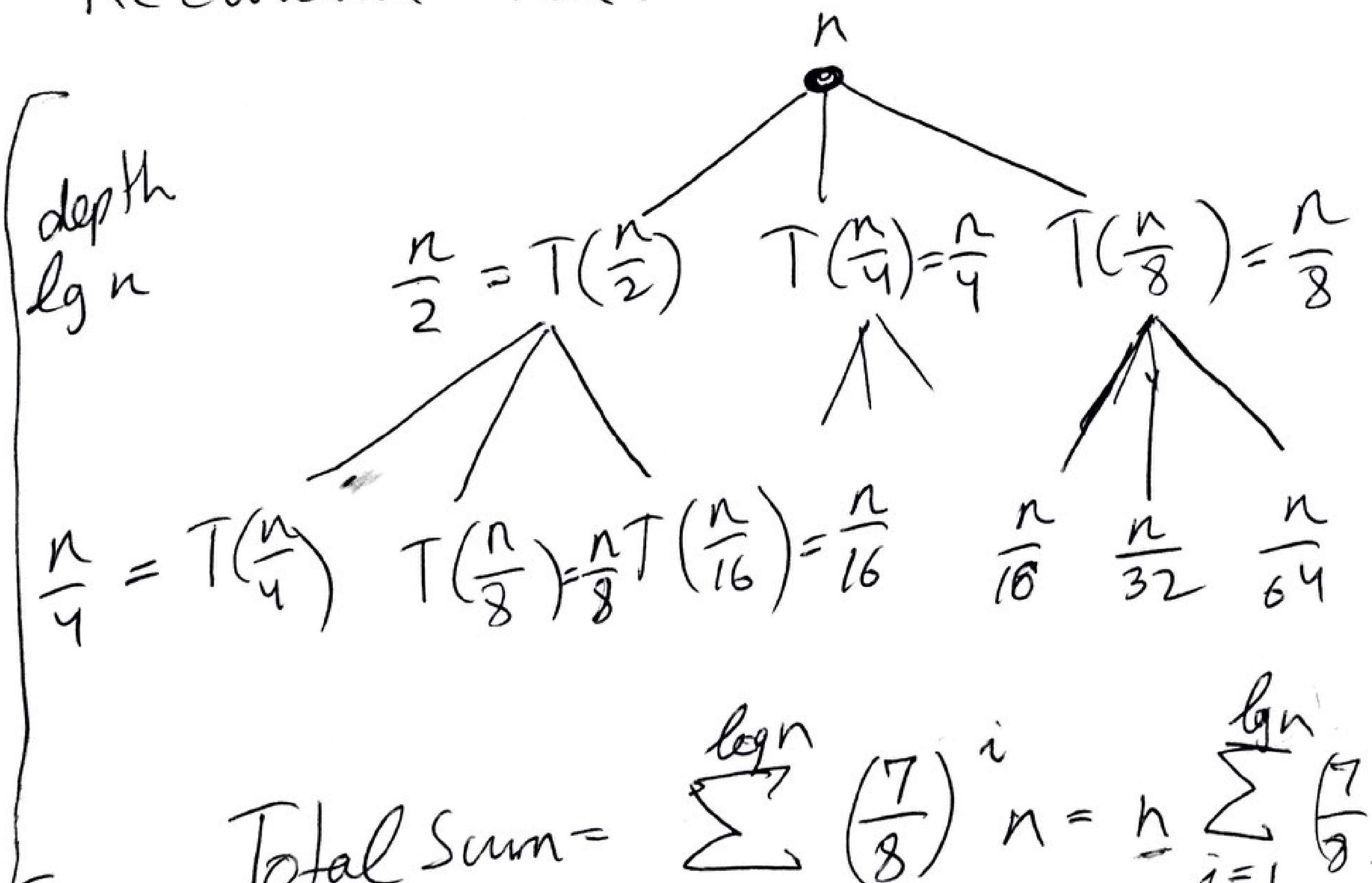
$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{4}\right) + T\left(\frac{n}{8}\right) + n$$

1. Guess. 2. Verify: Substitution Method.

Row Sum

Recursion Tree.



$\frac{7}{8} n.$

$\Rightarrow \left(\frac{7}{8}\right)^2 \cdot n.$

$$\text{Total Sum} = \sum_{i=1}^{\log n} \left(\frac{7}{8}\right)^i n = n \sum_{i=1}^{\lg n} \left(\frac{7}{8}\right)^i = O\left(\frac{7}{8}^{\lg n} \cdot n\right)$$

$$= \Theta(n) \ ?$$

$$\left(\frac{7}{8}\right)^{\lg n} = n^{\underbrace{\lg 7/8}_{<0}} \xrightarrow[n\to\infty]{} 0 \left(= O(1)\right)$$

$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{4}\right) + T\left(\frac{n}{8}\right) + n.$$

Lets Prove $T(n) = O(n)$, i.e. $T(n) \leq c \cdot n$ for some $c > 0$

IH. for all values $i < n$, $T(i) \leq c \cdot n$.

Ind. Step: Now prove that $T(n) \leq c \cdot n$.

$$T(n) \leq c\left(\frac{n}{2}\right) + \frac{cn}{4} + \frac{cn}{8} + n. \quad \boxed{\leq c \cdot n. \text{ need to show}}$$

$$= \frac{cn}{2} + \frac{cn}{4} + \frac{cn}{8} + n$$

$$= \frac{4cn}{8} + \frac{2cn}{8} + \frac{cn}{8} + n = \frac{7}{8}cn + n$$

$$= cn \underbrace{- \frac{cn}{8} + n}_{\text{residual.}}$$

needs to be ≤ 0

$$-\frac{cn}{8} + n \leq 0$$

$$n\left(1 - \frac{c}{8}\right) \leq 0.$$

$$c \geq 8$$

$T(n) \overset{?}{=} \Omega(n)$ want to show.

combine with $T(n) = O(n) \implies T(n) = \Theta(n)$

implies.

<u>Master Theorem</u> applies to most recurrences of the form:

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

e.g. $T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$ can be solved using Mast. Th.

$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{4}\right) + T\left(\frac{n}{8}\right) + n$ <u>cannot</u> be solved.

$T(n) = 2T(\lg n) + n^2$ <u>cannot</u> be solved.

$T(n) = T\left(\frac{n}{3}\right) + 1$ <u>can</u> be solved

$T(n) = T(n-1) + n$ <u>cannot</u> be solved.

$$T(n) = T\left(\frac{n}{2}\right) + \underbrace{n + n^3 + n! + 2^{n^{2^n}}}_{f(n)}$$

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

Compare $f(n)$ with $n^{\log_b a}$

Case 1 of the MT:

If $f(n) = \cancel{\Theta(n^{\log_b a})}$ ~~then~~ $= O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$

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

$f(n)$ grows polynomially slower than $n^{\log_b a}$

$T(n) = 4T(\frac{n}{2}) + n$.

$a = 4$, $b = 2$, ~~f(n) =~~

$f(n) = n$.

n vs. $n^{\log_2 4} = n^2$

Is $n = O(n^{2-\epsilon})$ for some $\epsilon > 0$?

yes: e.g $\epsilon = 1$

Then $T(n) = \Theta(n^{\log_b a}) = \Theta(n^2)$.

Case 2: If $f(n) = \Theta(n^{\log_b a} \lg^k n)$

for some constant $k \geq 0$

$\lg^k n = (\lg n)^k$ ~~$\lg(n^k)$~~

Then $T(n) = \Theta(n^{\log_b a} \lg^{k+1} n)$

$n \stackrel{?}{=} \Theta(n^2 \lg^k n)$ for some $k \geq 0$

Is $n \stackrel{?}{=} \Omega(n^2 \lg^k n)$ No!

$T(n) = 4T(\frac{n}{2}) + \underbrace{n^2}_{f(n)}$.

Is $n^2 = \Theta(n^2 \lg^k n)$ for some $k \geq 0$

Yes! Set $k = 0$

so $T(n) = \Theta(n^2 \lg n)$

Case 3.

If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some $\epsilon > 0$

Then. ~~$T(n) = \Theta($~~

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

There is a technical condition called "regularity condition". For this class, assume this condition always holds

Lets check $T(n) = 4T(\frac{n}{2}) + n^2$.

$n^2 \stackrel{?}{=} \Omega(n^{2+\epsilon})$ for $\epsilon > 0$

NO!

Lets check $T(n) = 4(T(\frac{n}{2})) + n^3$.

$n^3 \stackrel{?}{=} \Omega(n^{2+\epsilon})$ for $\epsilon > 0$.

Yes: e.g $\epsilon = 1$ or $\epsilon = 0.5$ or any $0 < \epsilon \leq 1$

$T(n) = \Theta(n^3)$

$T(n) = 4T(\frac{n}{2}) + \frac{n^2}{\lg n}$.

$f(n) = \frac{n^2}{\lg n}$ $\quad n^{\log_b a} = n^2$

Potentially Case 1: $\frac{n^2}{\lg n} \stackrel{?}{=} O(n^{2-\epsilon})$ for some $\epsilon > 0$ No

$n^d = \omega(\lg n)$ for all $d > 0$