Solving Recurrence Equations Asymptotically

Consider $T(n) = \begin{cases} 2T(n/2) + cn & n > 1 \\ c & n = 1 \end{cases}$

Assume for simplicity that n is a power of 2

Show $T(n) = cn(\log_2 n + 1)$

Base Case: $n = 1 \Rightarrow cn(\log_2 n + 1) = c = T(1)$ ✓

Inductive step: Assume $T(n) = cn(\log_2 n + 1)$ up to $m < n$

$$T(n) = 2T\left(\frac{n}{2}\right) + cn = 2\frac{cn}{2}\left(\log_2 \frac{n}{2} + 1\right) + cn$$

$$= cn(\log_2 n - 1 + 1) + cn = cn(\log_2 n + 1)$$

What if n is not a power of 2?

$$T(n) > T\left(2^{\lfloor \log_2 n \rfloor}\right) = c\,2^{\lfloor \log_2 n \rfloor}\left(\log 2^{\lfloor \log_2 n \rfloor} + 1\right)$$

$$\geqslant c\,2^{\log_2 n - 1}\left(\log_2 2^{\log_2 n - 1} + 1\right)$$

$$= \frac{cn}{2}\left(\log_2 n - 1 + 1\right) = \frac{1}{2} cn \log_2 n$$

Similarly:

$$T(n) < T\left(2^{\lceil \log_2 n \rceil}\right) = c\,2^{\lceil \log_2 n \rceil}\left(\log_2 2^{\lceil \log_2 n \rceil} + 1\right)$$

$$\leqslant c\,2^{\log_2 n + 1}\left(\log_2 2^{\log_2 n + 1} + 1\right)$$

$$= 2cn\left(\log_2 n + 2\right)$$

Both of these are $\Theta(n \log n)$

Since we are interested in asymptotic behavior, the rule of thumb is:

1) Ignore base case (it's satisfied for some large enough n)
   e.g. we could say $T(n) \le cn\log n$
   then $T(1)$ is **NOT** satisfied! (it's ok)

2) Ignore that n may not be a power of something appropriate, e.g. 2   treat $\frac{n}{2}$ as if it's an integer

3) Ignore floors and ceilings
   e.g. $T(n) = T(\lfloor \frac{n}{2} \rfloor) + T(\lceil \frac{n}{2} \rceil) + cn$
   is the real recurrence.

Substitution method:

Similar to induction with above 3 points in mind.

1) Guess solution

2) Verify it.

Example: $T(n) = 2T(n/2) + \theta(n)$

Guess $T(n) = \theta(n \log n)$ [how? Later ...]

Show upper and lower bounds separately by changing

$$T(n) = 2T(n/2) + cn$$

and showing:
- $T(n) \leq dn \log n$ for some d
- $T(n) \geq dn \log n$ for some d

- $T(n) \leq 2\left[d\frac{n}{2}\log\frac{n}{2}\right] + cn$ $\quad\left(T\left(\frac{n}{2}\right) \leq d\frac{n}{2}\log\frac{n}{2}\right)$

$$= dn\log\frac{n}{2} + cn = dn\log n - dn + cn$$

$$\leq dn\log n \quad \text{if} \quad -dn + cn \leq 0 \quad (d \geq c)$$

So $T(n) = O(n\log n)$ [upper bound]

- $T(n) \geq 2\left[d\frac{n}{2}\log\frac{n}{2}\right] + cn$ $\quad\left(T\left(\frac{n}{2}\right) \geq d\frac{n}{2}\log\frac{n}{2}\right)$

$$= dn\log\frac{n}{2} + cn = dn\log n - dn + cn$$

$$\geq dn\log n \quad \text{if} \quad -dn + cn \geq 0 \quad (d \leq c)$$

So $T(n) = \Omega(n\log n)$ [lower bound]

So $T(n) = \Theta(n\log n)$. Done!

$$T(n) = 8T(n/2) + \theta(n^2)$$

Guess $T(n) = \theta(n^4)$

$$T(n) = 8T(n/2) + cn^2$$

Show $T(n) \leq dn^4$

Assume $T\left(\frac{n}{2}\right) \leq d\left(\frac{n}{2}\right)^4$

$$T(n) \leq 8\, d\left(\frac{n}{2}\right)^4 + cn^2 = \frac{1}{2}dn^4 + cn^2$$

$$= dn^4 - \frac{1}{2}dn^4 + cn^2$$

$$\leq dn^4 \text{ if } -\frac{1}{2}dn^4 + cn^2 \leq 0$$

we need $-dn^2 + 2c \leq 0 \Rightarrow d \geq \frac{2c}{n^2}$

Any $d > 0$ is good for large enough n.

So $T(n) = O(n^4)$

Observe that we cannot prove $T(n) = \Omega(n^4)$ because we will need to find a constant $d \leq \frac{2c}{n^2}$ (we cannot)

In the previous example $T(n) = \Theta(n^3)$
That's why we can't prove $T(n) = \Theta(n^4)$

$$T(n) = 8T(n/2) + cn^2$$

Guess $T(n) = \Theta(n^3)$ and try to prove $T(n) \leq dn^3$

$$T(n) \leq 8\, d\left(\frac{n}{2}\right)^3 + cn^2$$

Assume $T\left(\frac{n}{2}\right) \leq d\left(\frac{n}{2}\right)^3$

$$= dn^3 + cn^2 \not\leq dn^3 \quad \text{(didn't work)}$$

<u>Note:</u> can't say $dn^3 + cn^2 = O(n^3)$, we have to prove exact form.

<u>Subtract a lower order term:</u> $T(n) \leq dn^3 - d'n^2$

$$T(n) \leq 8\left[d\left(\frac{n}{2}\right)^3 - d'\left(\frac{n}{2}\right)^2\right] + cn^2$$

$$= dn^3 - 2d'n^2 + cn^2$$

$$= dn^3 - d'n^2 \underbrace{- d'n^2 + cn^2}$$

works if $d' \geq c$.

How do we guess?

- Experience:

* $T(n) = 2T(\frac{n}{2} + a) + \theta(n)$ (where a is cte)

still guess $T(n) = \theta(n \log n)$

* $T(n) = 2T(\sqrt{n}) + \lg n$ let $m = \lg n$

$T(2^m) = 2T(2^{m/2}) + m$ let $T(2^m) = S(m)$

$S(m) = 2S(m/2) + m$

$S(m) = \theta(m \log m)$

$T(n) = \theta(m \log m) = \theta(\log n \cdot \log \log n)$

* $T(n) = 4T(n/2) + cn^2/\log n$

$T(n) = O(G(n))$, where $G(n) = 4G(n/2) + cn^2$

$T(n) = \Omega(F(n))$, where $F(n) = 4F(n/2) + cn^{2-\varepsilon}$

- Iterate:

$$T(n) = 8T(n/2) + cn^2$$

$$= 8\left[8T(n/4) + c\left(\frac{n}{2}\right)^2\right] + cn^2$$

$$= cn^2 + 2cn^2 + 64T(n/4)$$

$$= cn^2 + 2cn^2 + 64\left[8T(n/8) + c\left(\frac{n}{4}\right)^2\right]$$

$$= cn^2 + 2cn^2 + 4cn^2 + 64\left[8T(n/8)\right]$$

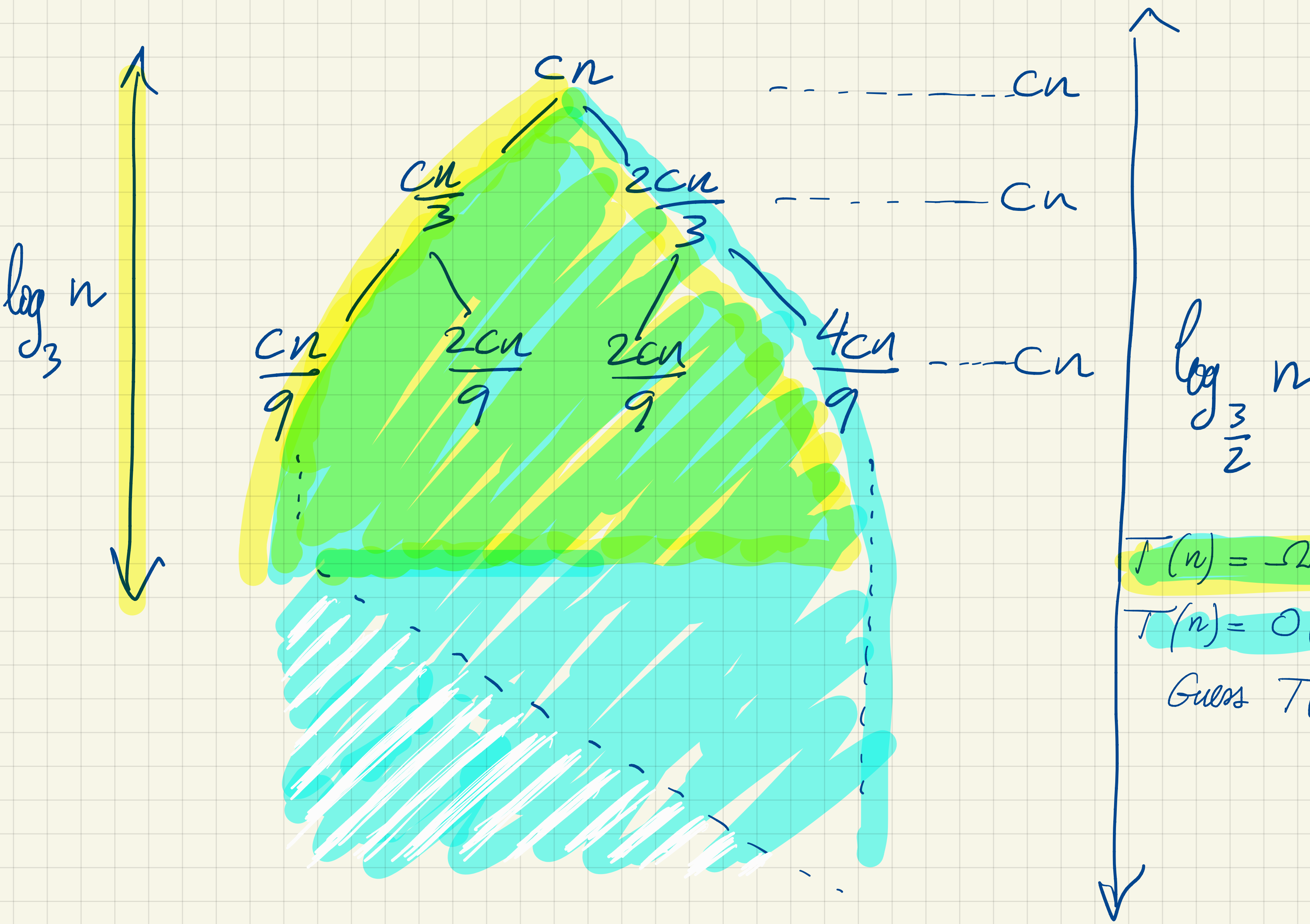
$$= cn^2 + 2cn^2 + 4cn^2 + \cdots + 8^{\log_2 n} T(1)$$

$$= cn^2 \sum_{i=0}^{\lg n - 1} (2^i) + \Theta(n^3)$$

$$= cn^2 \frac{2^{\lg n} - 1}{2 - 1} + \Theta(n^3) = \Theta(n^3)$$

- Recursive Tree visualization:

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



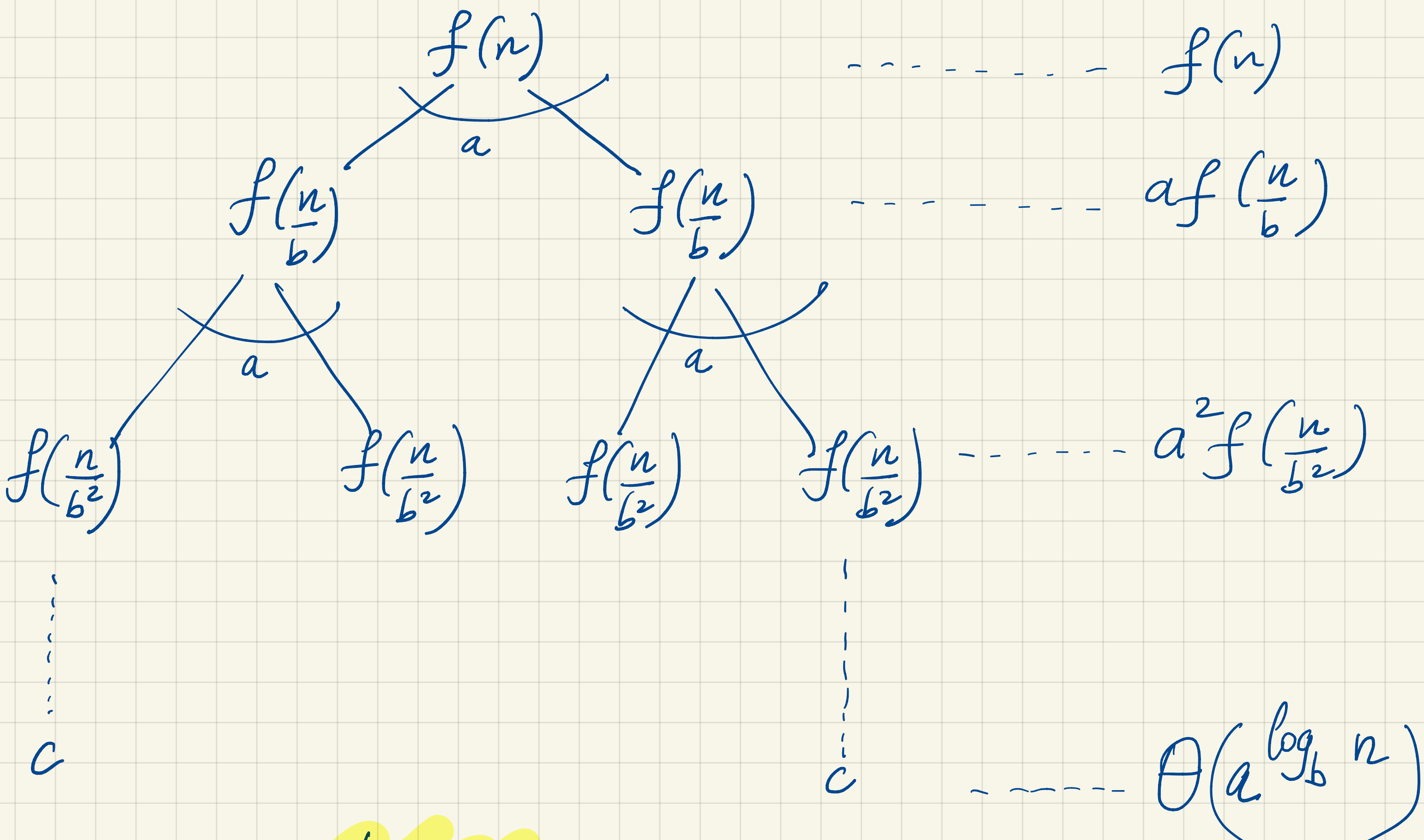
$T(n) = \Omega(n \log_3 n)$

$T(n) = O(n \log_{\frac{3}{2}} n)$

Guess $T(n) = \Theta(n \log n)$

Master method:

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(f(n)) \quad a \geq 1,\ b > 1,\ f(n) > 0$$



$$T(n) = \Theta\left(\sum_{i=0}^{\log_b n - 1} a^i f\left(\frac{n}{b^i}\right)\right) + \Theta\left(n^{\log_b a}\right)$$

proof ommited: Compare $f(n)$ to $n^{\log_b a}$

- $\dfrac{f(n)}{n^{\log_b a}} = O(n^{-\varepsilon})$ $\varepsilon > 0$, $T(n) = \Theta(n^{\log_b a})$

(leaves contribute more)

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

- $\dfrac{f(n)}{n^{\log_b a}} = \Omega(n^{\varepsilon})$ $\varepsilon > 0$, $T(n) = \Theta(f(n))$

(root contributes more)

technical condition:

$a f(\frac{n}{b}) \leq c f(n)$ for some $c < 1$

and sufficiently large n

e.g. True if $f(n) = n^k$

Examples:

- Merge Sort: $T(n) = 2T(n/2) + \Theta(n)$

  Case 2

  $n^{\log_b a} = n \qquad \frac{f(n)}{n} = \Theta(1) = \Theta(\log^0 n) \Rightarrow T(n) = \Theta(n \log n)$

- Strassen: $T(n) = 7T(n/2) + \Theta(n^2)$

  Case 1

  $n^{\log_b a} = n^{2.81}, \quad \frac{f(n)}{n^{2.81}} = n^{-0.81} \Rightarrow T(n) = \Theta(n^{2.81})$

- $T(n) = 4T(n/2) + n^3$

  Case 3

  $n^{\log_b a} = n^2, \quad \frac{f(n)}{n^2} = n^1 \Rightarrow T(n) = \Theta(n^3)$

- $T(n) = 4T(n/2) + \Theta(n^2/\log n)$

  $$n^{\log_b a} = n^2, \quad \frac{f(n)}{n^2} = \frac{1}{\log n} = \begin{cases} O(n^{-\varepsilon}), \varepsilon > 0 \ ? \\ \Theta(\log^k n), k \geq 0 \ ? \\ \Omega(n^{\varepsilon}), \varepsilon > 0 \ ? \end{cases}$$

  Can't use Master Method.

$T(n) = O(n^2 \log n)$, $T(n) = \Omega(n^{2-\varepsilon})$ for $\varepsilon > 0$

Guess $T(n) = n^2 \log \log n$ and use substitution method.