Recursion tree

این روش پیشتر در merge sort توضیح داده شده است، باید دقت شود:

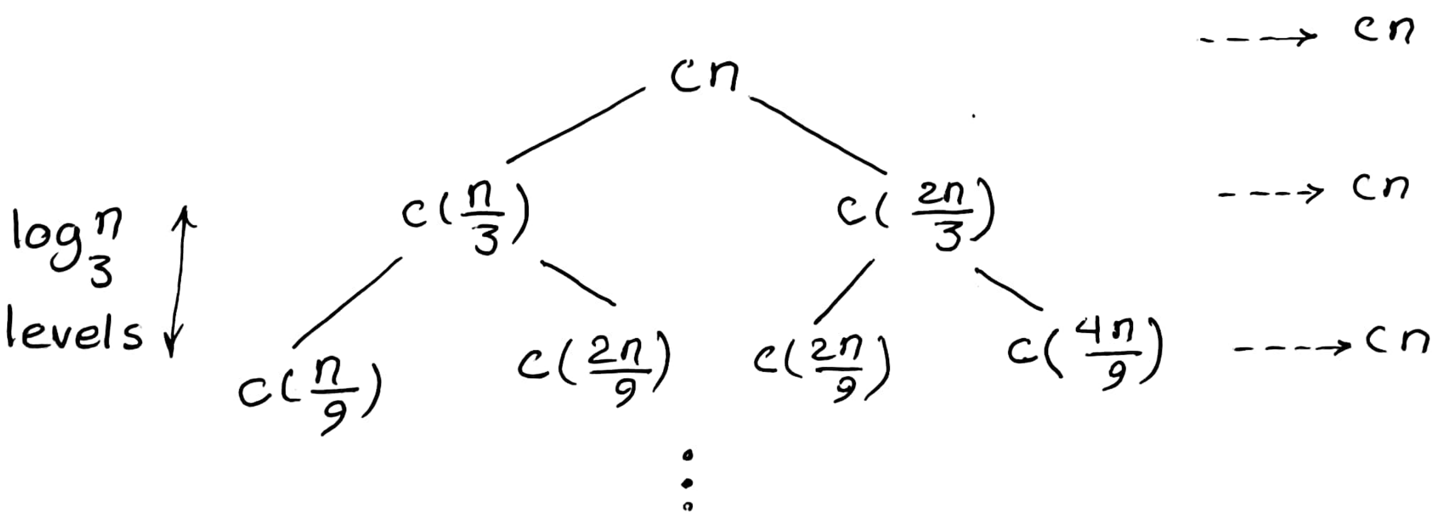
- این روش برای حدس solution برای یک recurrence است و نه اثبات.
- اثبات این حدس با substitution method است.
- این دو را باید باهم به کار برد.
- اینجا هم در صورت استفاده از asymptotic notation و θ باید هم upper و هم lower bound جداگانه محاسبه شود.

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

$$\Downarrow$$

for upper bnd. $T(n) \leqslant T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + cn$

for lower bnd. $T(n) \geqslant T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + cn$



حدس: Lower bound $\geqslant dn\log_3 n = \Omega(n\lg n)$

Upper bound $\leqslant dn\log_3 n = O(n\lg n)$

Master method

این روش مورد استفاده‌ی بسیار زیادی برای بدست آمدن recurrence های روش divide-and-conquer دارد.

فرض کنید یک recurrence به فرم زیر داریم:

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

$a \geq 1$

$b > 1$

$f(n) > 0$

براساس master theorem داریم:

ابتدا دو فرمول زیر را مقایسه کنید.

$$n^{\log_b^a} \quad \text{vs.} \quad f(n)$$

حالت اول: $f(n) = O(n^{\log_b^a - \varepsilon})$ برای ثابت $\varepsilon > 0$

به عبارت دیگر، $f(n)$ polynomially کوچکتر از $n^{\log_b^a}$ است.

$$\text{Solution: } T(n) = \Theta(n^{\log_b^a})$$

حالت دوم: $f(n) = \Theta(n^{\log_b^a} \lg^k n)$ که $k \geqslant 0$ است.

به عبارت دیگر $f(n)$ در poly log factor ی $n^{\log_b^a}$ است و نه کوچکتر

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

حالت دوم (حالت ساده‌تر)

$k=0 \Rightarrow$

$$f(n)=\theta(n^{\log_b^a}) \Rightarrow T(n)=\theta(n^{\log_b^a}\lg n)$$

حالت سوم: $f(n)=\theta(n^{\log_b^a+\varepsilon})$ برای $\varepsilon>0$ ،

regulary condition:

$\{$ $f(n)$ شرط زیر را ارضا کند: $af(\frac{n}{b})\leqslant cf(n)$ برای $c<1$ $\}$

به عبارت دیگر، $f(n)$ polynomially بزرگتر از $n^{\log_b^a}$ باشد.

Solution: $T(n)=\theta(f(n))$

مثال:

$$T(n)=5\,T(\tfrac{n}{2})+\theta(n^2)$$

$$n^{\log_2^5} \quad \text{vs.} \quad n^2$$

$\log_2^5-\epsilon=2$, for some constant $\epsilon>0$

$\rightarrow$ case 1:

$$T(n)=\theta(n^{\lg 5})$$

مثال:

$$T(n)=27\,T(\tfrac{n}{3})+\theta(n^3\lg n)$$

$$n^{\log_3^{27}}=n^3 \quad \text{vs.} \quad n^3\lg n$$

$\rightarrow$ case 2 with $k=1$ $\rightarrow$ $T(n)=\theta(n^3\lg^2 n)$

$$T(n) = 5T\left(\frac{n}{2}\right) + \theta(n^3)$$

$$n^{\log_2 5} \quad \text{vs.} \quad n^3$$

$$\lg 5 + \varepsilon = 3 \,, \quad \text{for constant } \varepsilon > 0$$

regularity condition (البته اگر $f(n)$ چندجمله‌ای باشد نیازی به چک نیست)

$$af\left(\frac{n}{b}\right) = 5\left(\frac{n}{2}\right)^3 = \frac{5n^3}{8} \leq cn^3 \quad \text{for } c = \frac{5}{8} < 1$$

$$\Rightarrow \text{case 3:} \quad T(n) = \theta(n^3)$$

مثال:

$$T(n) = 27\,T\left(\frac{n}{3}\right) + \theta\left(\frac{n^3}{\lg n}\right)$$

$$n^{\log_3 27} = n^3 \quad \text{vs.} \quad \frac{n^3}{\lg n} = n^3 \lg^{-1} n \neq \theta(n^3 \lg^k n)$$

$$\text{for } k \geq 0$$

استفاده از master method ممکن نیست.

نکته: در این course، master theorem را اثبات نخواهیم کرد.

حالت های غیر قابل قبول برای استفاده از master method به تفکیک:

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

۱- $T(n) = 2^n T\left(\frac{n}{2}\right) + n^n$ ، چون a مقدار ثابت نیست.

۲- $$T(n) = 2T\left(\frac{n}{2}\right) + \frac{n}{\log n}$$

چون اختلاف $f(n)$ و $n^{\log_b^a}$ غیرچندجمله ای است.

۳- $$T(n) = 0.5\, T\left(\frac{n}{2}\right) + n$$

$a<1$ است درحالی که هرمسئله نمی تواند کمتر از یک زیرمسئله داشته باشد.

۴- $$T(n) = 64\, T\left(\frac{n}{8}\right) - n^2 \lg n$$

$f(n)$ مثبت نیست. (در واقع $f(n)$ همان combination time است که باید time مثبت باشد)

۵- $$T(n) = T\left(\frac{n}{2}\right) + n(2 - \cos n)$$

حالت سوم ونقض regularity condition (regularity violation)

Akra-Bazzi method :

روشی برای تحلیل asymptotic فرمول های recurrence که از روش divide-and-conquer بدست می آیند وقتی که سایز subproblem ها به طور اساسی باهم متفاوت باشد.

- تعمیم master theorem است برای divide-and-conquer

فرم کلی recurrence مورد استفاده :

$$T(x) = g(x) + \sum_{i=1}^{k} a_i T\left(b_i x + h_i(x)\right) \quad \text{for } x \geq x_0$$

شرایط مورد استفاده در این روش :

- base case های مورد نیاز تعریف شده باشد.
- برای همه i ها ؛ به ازای هر i ، a_i و b_i ثابت هستند.
- برای هر i ؛ $a_i > 0$
- برای هر i ؛ $0 < b_i < 1$
- برای ثابت c ؛ $|g(x)| \in O(x^c)$
- برای هر i ؛ $|h_i(x)| \in O\left(\frac{x}{(\log x)^2}\right)$
- x_0 ثابت است.

$T(x)$ به مقدار متغیری به نام p بستگی دارد که باید از رابطه زیر بدست آید .

$$\sum_{i=1}^{k} a_i b_i^{p} = 1$$

- p رابطه بالا را برقرار می کند .

Solution of recurrence :

$$T(x) = \Theta\left(x^p\left(1 + \int_1^x \frac{g(u)}{u^{p+1}}\,du\right)\right)$$

یک مثال:

$$T(n)=\begin{cases} 1 & 0\leq n<3 \\ n^2+\frac{7}{4}T\left(\left\lfloor \frac{1}{2}n\right\rfloor\right)+T\left(\left\lceil \frac{3}{4}n\right\rceil\right) & n>3 \end{cases}$$

$$p=? \qquad \frac{7}{4}\left(\frac{1}{2}\right)^p+\left(\frac{3}{4}\right)^p=1 \Rightarrow p=2$$

$$T(x)\in\theta\left(x^p\left(1+\int_1^x \frac{g(u)}{u^{p+1}}du\right)\right)$$

$$=\theta\left(x^2\left(1+\int_1^x \frac{u^2}{u^3}du\right)\right)$$

$$=\theta\left(x^2(1+\ln x)\right)$$

$$=\theta\left(x^2\lg x\right)$$

مثال ساده‌تر $T(n)=2T(\frac{n}{2})+n-1$ که قبلاً برای merge sort، master method حل شده است، که البته دقیق‌تر آن به این صورت است.

$$T(n)=\begin{cases} T\left(\left\lfloor \frac{1}{2}n\right\rfloor\right)+T\left(\left\lceil \frac{1}{2}n\right\rceil\right)+n-1 & n>1 \\ 0 & n=1 \end{cases}$$

را با Akra-Bazzi حل می‌کنیم.

نتیجه نهایی: $\theta(n\lg n)$