

طرایی الگوریتم :

زیرین ها :

introduction to algorithms CLRS 3rd edition - 7

fundamental of computer algorithm ; horowitz - 2

: بامباردی

تمرين 3 میانسیم 7 پایانسیم 10

زیرین ها :

مهندسی و دستی الگوریتم ها -

مروری بر روش توابع -

صرری بر روابط بازنیتی : درفت بازنیت و قصبه اصلی

مرتب سازی معنی (Selection problem) : ابیات رسمی و تکلیف زمانی

مرتب سازی حقیقی : مرتب سازی زمانی و ابیات رسمی

(selection problem) : مسأله سایر اثبات صیانه و کاری عو در درین آور

تکلیف سریع -

(divide and conquer) دو خانه -

مسأله حساب ماتریس ها و مرتب اسیدان

برنامه نویسی بویا -

الگوریتم های حدسینه : مسأله کره هامون و کوه هیئتی

(backtrack) روش عبور

(branch & bound) روش تا خود

مسأله کرتاهیتین مسیر : dikstrom , floyd marshal

maximum flow آلگوریتم های جریان بینی

نفری و بیخیلی محاسبات

طراحی الگوریتم : حلبه اول

متدها - جایه اگوریتم

تعیین الگوریتم : دنباله ای از ترتیب متعاقباً باید اجرا شود. و ترجیح دادن مخصوصاً در رویب مسئله را باعث می کند
تعیین مسئله : هر زدج مرتب متعلق از ترتیب ورودی و خروجی مسئله می باشد: مسئله مرتب سازی
اهمیت طراحی الگوریتم : بینی بعیی از مسائل دنیای واقعی الگوریتم نداریم و باید طراحی نظر

نمایل:

$$n = 10^6 \text{ تعداد اعنای اولی}$$

مسئله مرتب سازی:

با وجود درجه تردید n پیشتر از زمان حل
مسئله A برابر کنند از $\Theta(n^2)$ بودهون الگوریتم A
آن تردید و این اهمیت بینیه بون الگوریتم راستایی $O(n \log n)$

A درجه:
 10^9 INS/sec / کامپیوت

الگوریتم: $O(n^2)$

$$\frac{\text{ویندهی رقانی}}{\text{سرعت کامپیوت}} = \frac{(10^6)^2}{10^9} = 10^3 \text{ sec}$$

B درجه:
 10^7 INS/sec / کامپیوت

الگوریتم: $O(n \log n)$

$$\frac{\text{زمان حل}}{10^7} = \frac{10^6 \times \log 10^6}{10^7} = 2 \text{ sec}$$

تحليل زمانی و ایندیکات درست $\Omega(n^2)$ درجی / insertion sort

Insertion Sort (A, n)

بيان الالعارات وسبلها

فَرَفِنْ حَوَّلَتْمُ اِيْنِدْ لَسْ اَوْلَارِيَهْ لَاتْ

1. for $j=2$ to n
 2. Key = $A[j]$
 3. $i=j-1$
 4. while (Key < $A[i]$ and $i>0$)
 5. $A[i+1] = A[i]$ } shifting elements to the right
 6. $i=i-1$
 7. $A[i+1] = \text{Key}$

: incremental روش

الآن نستعرض الـ incremental insertion sort (الـ جبر)

با هم بار بسیاری خوبی داشتند از آن خانه را تازه تر بازگردند و سمت پشت خانه را خود بسیاری تغیر دادند. از آن خانه را تازه تر بازگردند و سمت پشت خانه را خود بسیاری تغیر دادند.

اپنے سمجھیں:

لذاره را باید خود طبق درستگاه تعیین کرد. درستگی آن را اثبات کنیم.

نایاب کیملتہ برائی زیر آرائی [1- j] و $A[1-j+1..n]$ insertion sort : در اینجا مامرا میں حالت for زیر آرائی A [1- j] و $A[1-j+1..n]$ میں مرتب نہیں است.

حالت پنجم:

برای اینکه در حالتی که $A[i:j]$ مقداری از A باشد، برای هر $i < j$ از $A[i:j]$ زیرآرایی است که عناصر اولی از A را از i -مین تا $j-1$ -مین عناصری از A تشکیل می‌دهند.

طراحی الگوریتم: حلبة (ROM)

تکلیف زمانی

زمان اجرایی بینایی هم به وسیله ای الگوریتم و عدم بقدرت پردازشی مانند که آن را اجرا نمی کند بستگی دارد (عمل محاسباتی)

The RAM model: Random Access Model

کامپیوتر تواند در هر یک دستور را می تواند اجرا کند، دستورات مانند $\{$ جمع، رسانید، ضرب و تقسیم متعالیه shift $\}$ بازیگردانی می کنند و بازیگردانی می کنند دستور را باید است.

تابع زمان اجرایی الگوریتم:

تابعی از تعداد اعداد و روشی است.

insertion sort :

$$1. C_1 \times n \quad (2 \text{ to } n+1)$$

$$2. C_2 \times (n-1)$$

$$3. C_3 \times (n-1)$$

$$4. C_4 \times \sum_{j=2}^n (t_{ij} + 1)$$

$$5. C_5 \times \sum_{j=2}^n t_{ij}$$

$$6. C_6 \times \sum_{j=2}^n t_{ij}$$

$$7. C_7 \times (n-1)$$

حالة مبارڪه از بدنهای اجرایی در header

best case:

$$t_{ij} = 0 \rightarrow T(n) = an + b$$

worst case:

$$\begin{aligned} t_{ij} = j-1 &\rightarrow T(n) = C_1 n + C_2 (n-1) + C_3 (n-1) \\ &+ C_4 \sum_{j=2}^n j + (C_5 + C_6) \sum_{j=2}^n (j-1) + C_7 (n-1) \\ &= \frac{(n+2)(n-1)}{2} \qquad \qquad \qquad \frac{n(n-1)}{2} \\ &= an^2 + bn + c \end{aligned}$$

$$\text{insertion sort} = \mathcal{L}(n) = O(n^2)$$

برای ما زمان اجرایی حالت مترقبه اهمیت دارد که بعد از مرحله آن می پردازیم.

بار تدریس:

امید ریاضی:

$$\bar{x} = E[x] = \sum_x x \cdot P\{X=x\}$$

از طریق صیغه‌نمایی: $E[ax+by] = a \cdot E[x] + b \cdot E[y]$ امید ریاضی تولید علی = ترکیب حفظ امید ریاضی

متغیر تصادفی چنین (Indicator random variable)

متغیری که باید رخداد انتساب می‌باشد و فقط می‌تواند مقادیر ۰ و ۱ را اختیار کند

مثال باید رخداد A داریم:

$$I(A) = \begin{cases} 1 & \text{رخداد A} \\ 0 & \text{رخداد ندارد A} \end{cases}$$

: insertion sort مترادف

از آنجایی که سیویجی زمانی این الگوریتم به تعداد اجرایی حلقة t_j (تکمیلی) را در بازه امید ریاضی t_j را بیابیم.

$$0 \leq t_j \leq j-1 \rightarrow E[t_j] = \sum_{k=0}^{j-1} k \cdot P\{t_j=k\}$$

برای محاسبه امید ریاضی از این روش اول باید احتمال اینکه key از همین عرضی بزرگتر نباشد، سپس اینکه فقط از تکمیلی عدد، فقط از در بعد از آنچه بزرگتر باشد را محاسبه کنیم و در رابطه قدرت داشیم \leftarrow محاسبات سخت و زمانی تر

$$x_i = \begin{cases} 1 & \text{کوچکتر از } A[i] \text{ و } A[i] \text{ کوچکتر از } A[j] \\ 0 & \text{وگرنه} \end{cases} \rightarrow t_j = \sum_{i=1}^{j-1} x_i$$

$$E[t_j] = E \left[\sum_{i=1}^{j-1} x_i \right] = \sum_{i=1}^{j-1} E[x_i] = \sum_{i=1}^{j-1} 1/2 = 1/2 (j-1)$$

$$\rightarrow E[x_i] = 0 \times P\{A[i] < \text{key}\} + 1 \times P\{A[i] > \text{key}\} = P\{A[i] > \text{key}\} = 1/2$$

* نتیجه: از آنجایی که در حالت مترادف $t_j = j-1$ و در بیشترین حالت $t_j = j-1 - \frac{j-1}{2}$ از عکاظ سیویجی زمانی

$$T(n) \in \Theta(n^2)$$

حالات مترادف های بیشترین حالت است \leftarrow

طایف الگوریتم : حلبه سر

طایف زمانی را بسته بررسی : Merge Sort

start index
↑
end index
mergesort(A, i, j)

1. if $i < j$:

2. $q = (i+j)/2 \rightarrow \text{ماں } ①$

3. mergesort(A, i, q) } $\text{ماں } ②$

4. mergesort($A, q+1, j$) } $\text{ماں } ③$

5. merge(A, i, q, j) $\rightarrow \text{ماں } ④$

ایت الگوریتم 3 مام را درد :

① مام تعقیب بزیر مسائل

② مام حل بازگشتی زیر مسائل

③ مام ترتیب جواب بزیر مسائل

: نہیں

merge(A, i, q, j)

$n_1 = q - i + 1, n_2 = j - q \rightarrow \text{سایز 2 زیر مسائل}$

$L[n_1+1] = \infty, R[n_2+1] = \infty$

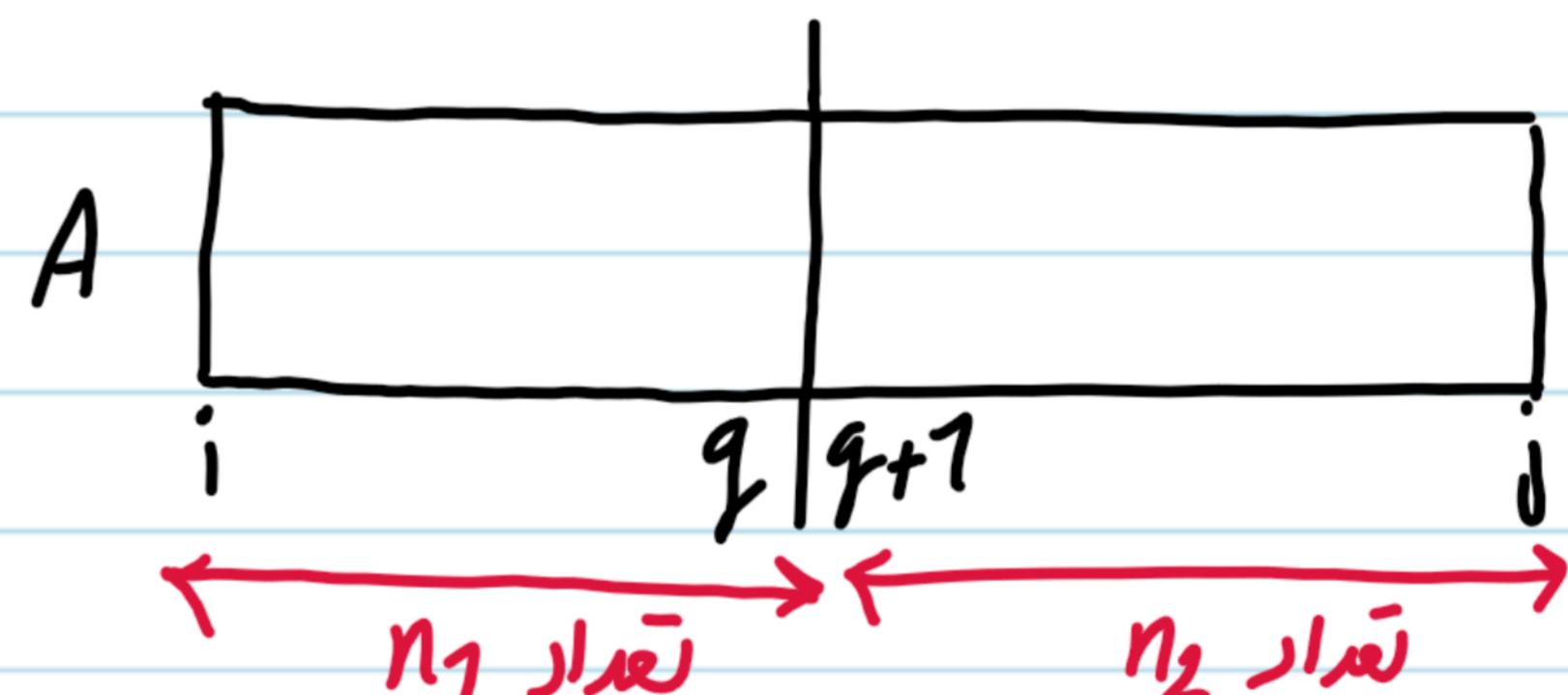
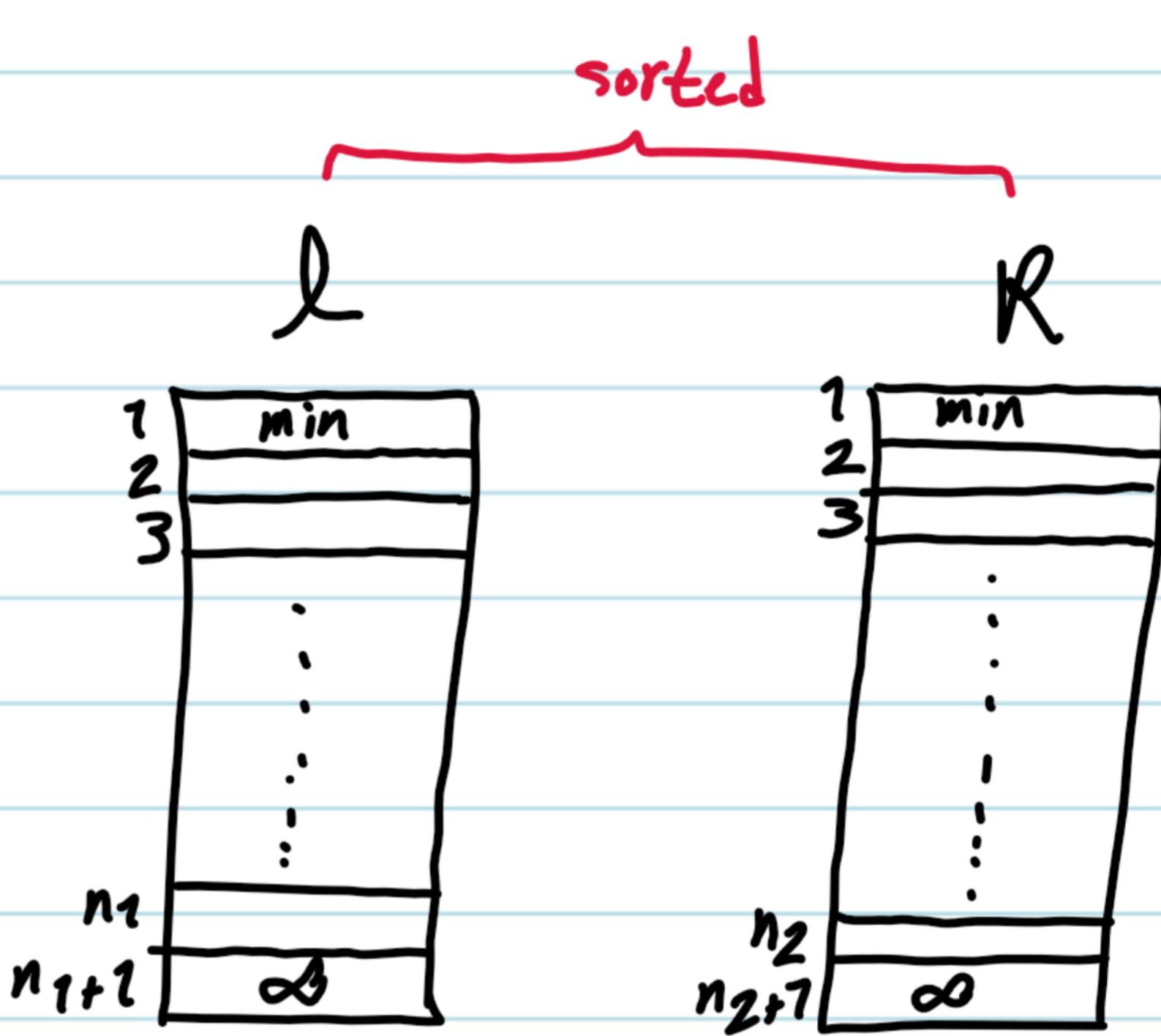
for $l=1$ to n_1 :

$L[l] = A[i+l-1]$

کوئی کوتون اور زیر آرائی
مرتب نہ رہا در اد آرائی جملان

for $r=1$ to n_2 :

$R[r] = A[q+r]$



$l=r=1$

for $k=i$ to j :

if $L[l] < R[r]$

$A[k] = L[l]$
 $l++$

else

$A[k] = R[r]$
 $r++$

متابیہ عنصر مینم در آرایه ر

پھر کوتون مینم کو جوینٹ در آرایه نہیں

ابات درست الوریتم merge :

فرض: آرایه دردی A از n تا و دارای $L+R$ تا مجموعه ای است.

حکم: در اینجا الوریتم آرایه A از n تا مجموعه ای است.

ثابت حلقه (loop invariant): (در اینجا λ کام مطلعه for پایانی، زید آرایه $A[i, \dots, k-1]$ شامل $i-K$ تا i کوچکترین عناصر آرایه $R[r, L[l], \dots, n+1]$ و $R[n+1, L[l+1], \dots, n+L]$ به صورت صرتیز صوری است و همین $R[r, L[l], \dots, n+1]$ کوچکترین عناصر آرایه A هستند.)

پایه استعداد: ($i=r=1$ و $k=i$): (در اینجا λ کام مطلعه for پایانی، زید آرایه $A[i-1, \dots, i]$ شامل 0 تا $i-1$ کوچکترین عناصر آرایه L و R بصرت صرتیز صوری است. $R[T]$ کوچکترین عناصر آرایه های A هست که در A حضور داشته باشند.)

نام لذار: فرض کنیم ثابت حلقه برای K بدقة از است و برای $K+1$ اینست کنیم
طبق فرضی، $L[r, L[l], \dots, R[r]]$ کوچکترین عناصر L و R هستند که در A نباشند. سپس اگر $\min(A[r, L[l], \dots, R[r]]) < \min(A[r, L[l], \dots, R[r-1]])$ باشد، $A[r, L[l], \dots, R[r]]$ کوچکترین عناصر آرایه $A[r, L[l], \dots, R[r-1]]$ هستند. و حین آن $L[r, L[l], \dots, R[r]]$ کوچکترین عناصر L و R صرتیز صوری باشند.

پس از $L[r, L[l], \dots, R[r]]$ کوچکترین عناصر L و R هستند. مینیموم آرایه $A[r, L[l], \dots, R[r]]$ برابر با $\min(A[r, L[l], \dots, R[r-1]])$ است. پس این $A[r, L[l], \dots, R[r]]$ کوچکترین عناصر آرایه $A[r, L[l], \dots, R[r-1]]$ هستند. و درنتیجه الوریتم $Merge$ درست است.

الجواب : mergesort (رسی الورتیم)

: merge جملہ

$$\underbrace{a \cdot n + b}_{\text{حالة for بابان}} \rightarrow O(n)$$

$\leq C_1 + \max \{C_2, C_3\} (n_1 + n_2) + C_4 n$

دو حلقة for اولیه زمان $C_3 n_2$ و دیگری زمان $C_2 n_1$

کل مجموع زمان $C_1 + C_4 n$

کمتر از $C_1 + C_4 n + C_2 n_1 + C_3 n_2$

کمتر از $C_1 + C_4 n + C_2 n + C_3 n$

کمتر از $C_1 + C_4 n + C_1 n = C_1 + C_4 n + C_1 n = C_1 + C_4 n$

کمتر از $C_1 + C_4 n$

if $C_2 = C_3 = C_4 = 1$ $C_1 = 0 \rightarrow n \leq \underline{\text{in fact}} \rightarrow \mathcal{O}(n) \rightarrow \text{merge} = \Theta(n)$

١- ترتيب زippy : Merge Sort

$$\begin{aligned} T(n) &= C_1 + C_2 + T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) \\ &= T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) \\ &= 2 \underbrace{T(n/2)}_{\text{از بیان از در زمانی با } \lfloor n/2 \rfloor} + \Theta(n) \end{aligned}$$

* ارائے کام (زمانی بعد ابریکی میلے) *

* اردر زمانی merge sort در بین ریدترین حالت بکار رفته است.

حاجی الگوریتم: حلبچهارم

میرایی در رشد توابع

$$1. f(n) = O(g(n)) \Leftrightarrow \exists C_0 > 0, n_0 > 0 \mid \forall n > n_0 \quad f(n) \leq C_0 \cdot g(n)$$

عنی رشد تابع f بسیار پایه‌ساوی رشد تابع g است (عنی):

$$\lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} = C_0$$

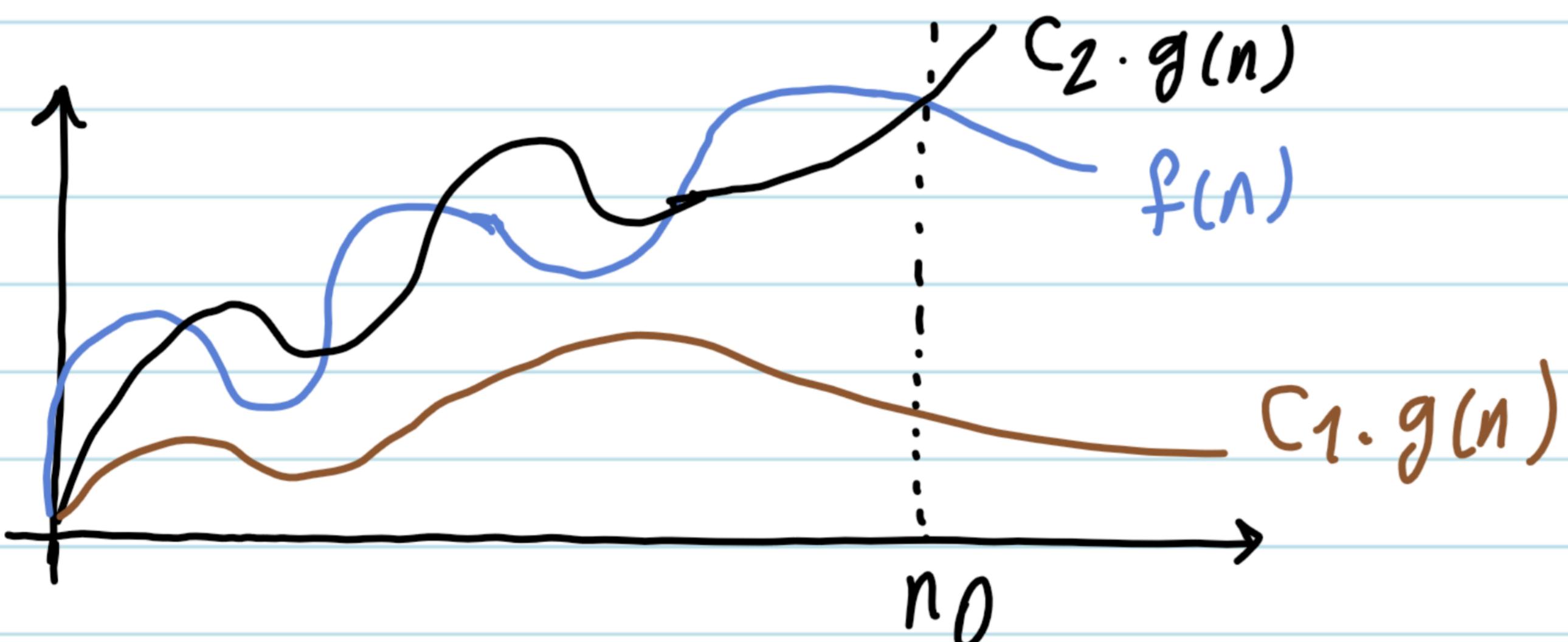
$$2. f(n) = \Omega(g(n)) \Leftrightarrow \exists C_0, n_0 > 0 \mid \forall n > n_0 \quad f(n) \geq C_0 \cdot g(n)$$

عنی رشد تابع f از تابع g کمتر پایه‌ساوی باشد (عنی):

$$\lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} = C \in \mathbb{Z}$$

$$3. f(n) = \Theta(g(n)) \Leftrightarrow f(n) = O(g(n)) \quad \& \quad f(n) = \Omega(g(n))$$

$$\Leftrightarrow \exists C_1, C_2, n_0 > 0 \mid \forall n > n_0 \quad C_1 \cdot g(n) \leq f(n) \leq C_2 \cdot g(n)$$



الگوریتم در بلندین طبقات از لدر n^2 و در بینین طبقات از
دستور می‌باشد $T(n) = O(n^2) = \Omega(n)$

در بینین طبقات $T(n) \in \Theta(n)$ $\xrightarrow{\text{جتن}} : C_1 \cdot n \leq T(n) \leq C_2 \cdot n$

در بلندین طبقات $T(n) \in \Theta(n^2)$ $\xrightarrow{\text{جتن}} : C_3 \cdot n^2 \leq T(n) \leq C_4 \cdot n^2$

طراحی الگوریتم: حلبة دفعی

4. $f(n) \in O(g(n)) \iff \forall c > 0, \exists n_0 > 0 \mid \forall n \geq n_0 \quad f(n) \leq c \cdot g(n)$

$$\lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} = \infty$$

معنی رسمی: $f(n)$ و $g(n)$ از تابع دivergent هستند.

5. $f(n) \in \omega(g(n)) \iff \forall c > 0, \exists n_0 > 0 \mid \forall n \geq n_0 \quad f(n) \geq c \cdot g(n)$

$$\lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} = 0$$

معنی $f(n)$ کمتر از $g(n)$ است.

معنی $g(n)$ حد پایی دارد / محدودی از تابع لغتنمی دارد / از n بزرگ شود: $\omega(g(n))$ است.

$$T(n) = f(n) + O(f(n)) \rightarrow T(n) \in \Theta(f(n))$$

حل روابط بازگشتی

برای تحلیل زمانی الгорیتم‌های divide & conquer به استرسک نیاز داریم.

استرسک درخت بازگشت
(Recursion tree)

استرسک‌های حل روابط بازگشتی

استرسک قضیه اصلی
(master theorem)

$$\begin{aligned} \Theta(1) \\ 2T(n/2) \\ \Theta(n) \end{aligned}$$

} قسم ۱ - حل بازگشتی

} قسم ۲ - حل بازگشتی

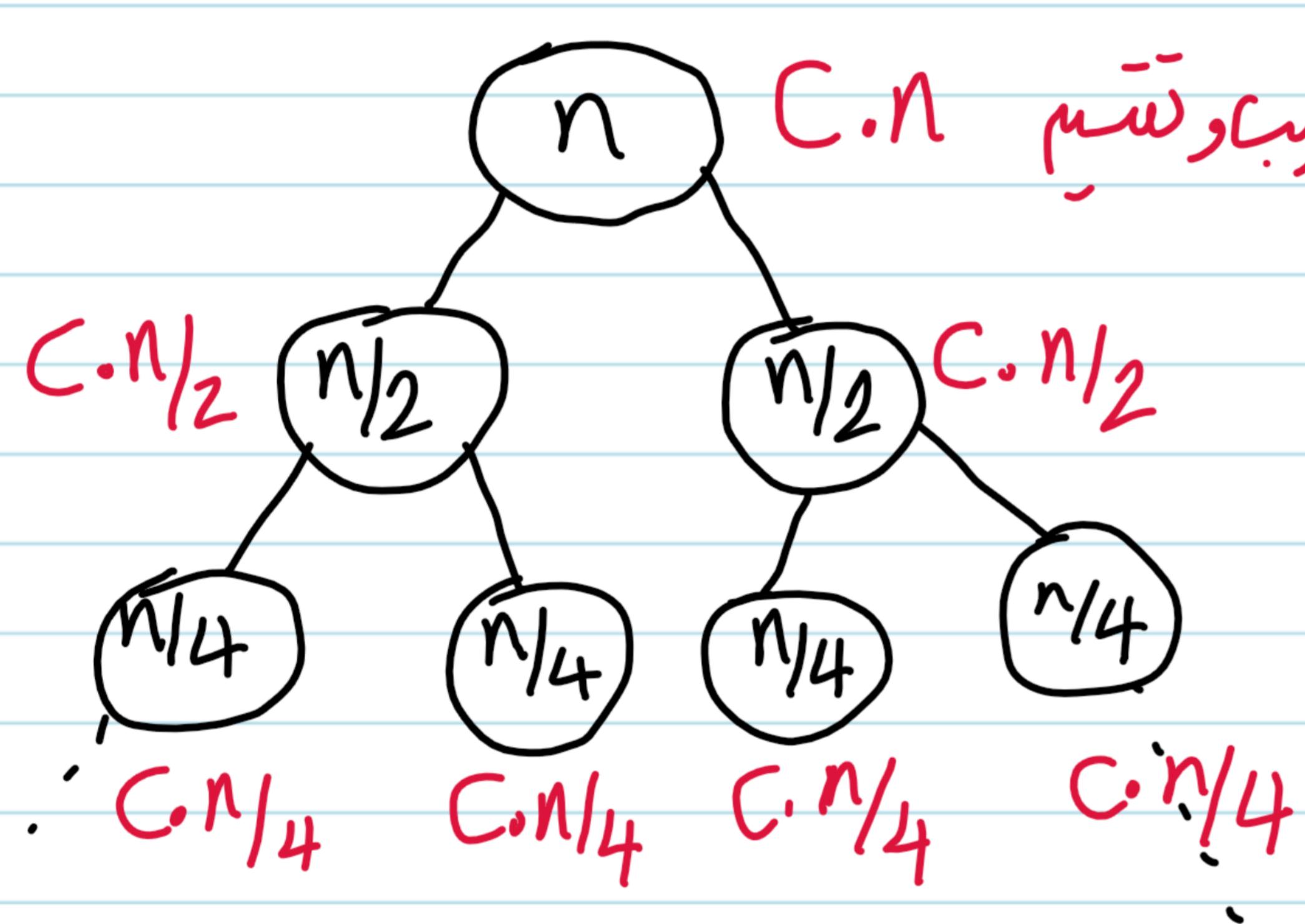
} قسم ۳ - مرتب

$$T(n) = 2T(n/2) + \Theta(n) \quad \leftarrow \text{merge sort : مدل.}$$

$$T(1) = C'$$

C.n هزینه ترکیب و تقسیم

هزینه کل = جمع کل هزینه‌های W رله‌ها نزدیک



$$i = \log n + 1 \leftarrow \frac{n}{2^{i-1}} = 1 \leftarrow 1 = \text{عدد رله مجموعی.}$$

نمودار سطر

1
2
3
⋮
i = ?

هزینه سطر

$$Cn$$

$$2 \times Cn/2 = Cn$$

$$4 \times Cn/4 = Cn$$

$$\vdots$$

$$Cn$$

سایز سطر

$$n$$

$$n/2$$

$$n/4$$

$$n/2^{i-1}$$

تعداد رله‌ها

$$1$$

$$2$$

$$4$$

$$\vdots$$

$$2^{i-1}$$

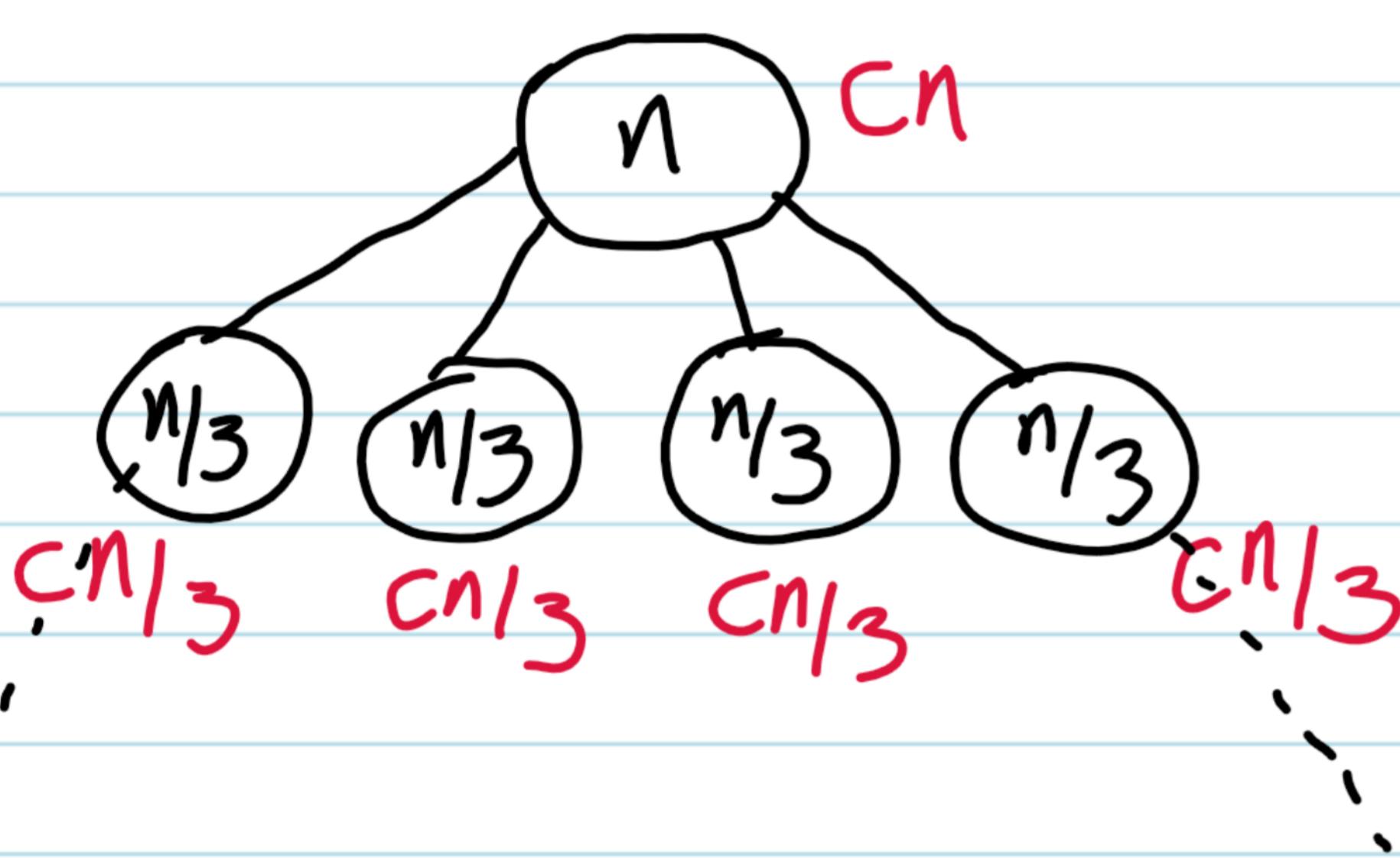
$$\text{هزینه مل} = \underbrace{\text{هزینه جریان های کل}}_A + \underbrace{\text{هزینه تعداد بدرها}}_B = cn \log_2^n + cn \in \Theta(n \lg n)$$

$$A = \log_2^n \times cn$$

$$B = \text{تعداد بدرها} \times \text{هزینه بدر} = 2^{i-1} \times c' = 2^{\log_2^n + 1 - 1} \times c' = 2^{\log_2^n} \times c' = n \times c' = n \cdot c'$$

$$T(n) = 4T(n/3) + \Theta(n), \quad T(1) = c'$$

: حل



$$\begin{aligned} \text{هزینه} &= \sum_{i=1}^{\log_3^n} (4/3)^{i-1} cn + 4^{i-1} \\ &= \sum_{i=1}^{\log_3^n} (4/3)^{i-1} cn + n \log_3 4 \end{aligned}$$

	هزینه مل	هزینه تعداد بدر	تعداد بدر	سایز سطح
1	cn	1	n	
2	4cn/3	4	n/3	
3	16cn/9	16	n/9	
:	:	:	:	
i	$(4/3)^{i-1} cn$	4^{i-1}	$n/3^{i-1}$	

$$\frac{1}{3^{i-1}} = 1 \rightarrow n = 3^{i-1}$$

$$i = \log_3 n + 1$$

$$\text{هزینه بدرها} = \sum_{i=1}^{\log_3^n} (4/3)^{i-1} cn = \frac{((4/3)^{\log_3^n} - 1)}{4/3 - 1} \cdot cn \in \Theta(n^{1.26})$$

$$\text{هزینه اخرين} = 4^{i-1} = 4^{\log_3^n + 1 - 1} = n^{\log_3 4} = n^{1.26} \cdot c'$$

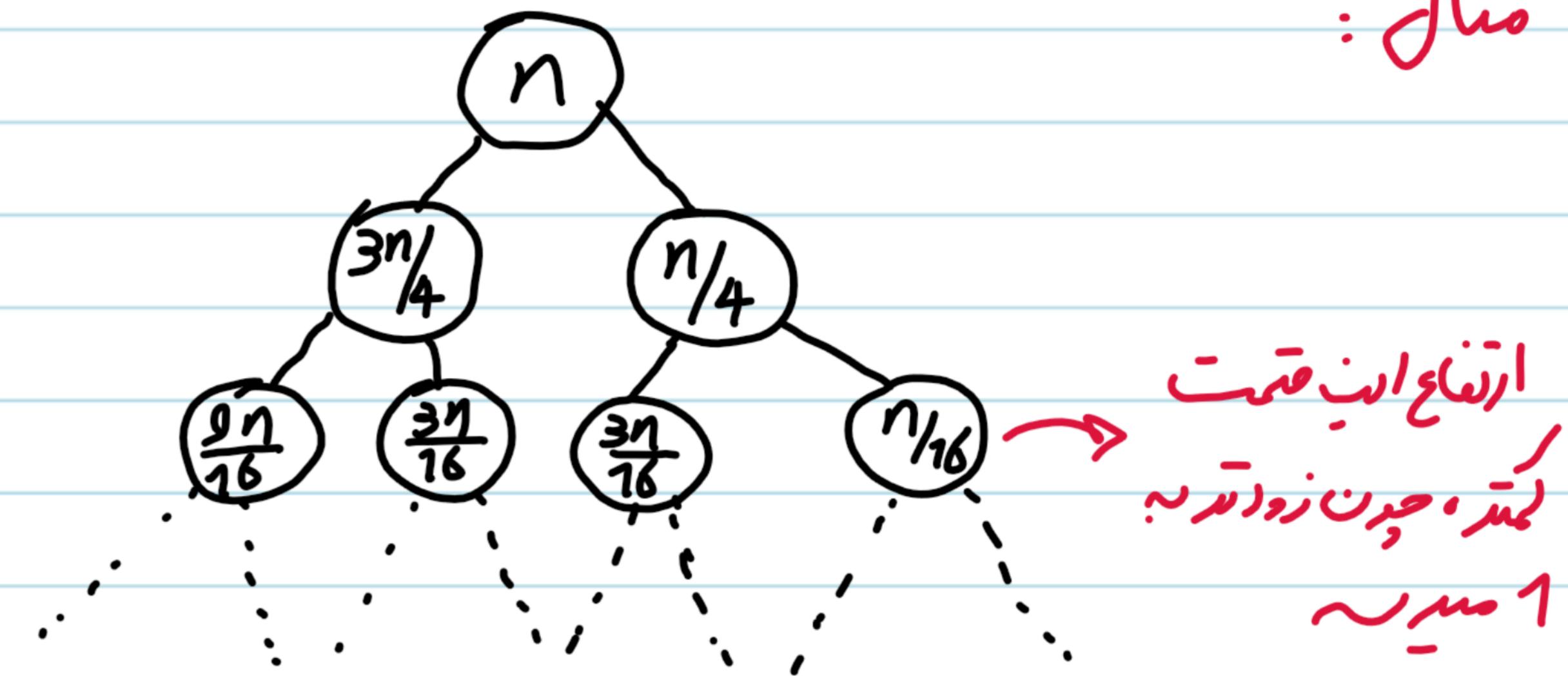
$$\text{هزینه مل} = c'n^{1.26} + \Theta(n^{1.26}) \in \Theta(n^{1.26})$$

طایی الگوریتم: حلبه سُم

$$T(n) = T(n/4) + T(3n/4) + \Theta(n)$$

$$T(1) = C'$$

او زیر مسئله با سایز متفاوت



هزینه سعر سماره سطر

$$1 \quad cn$$

$$2 \quad \frac{cn}{4} + \frac{3cn}{4} = cn$$

$$3 \quad 2 \times \frac{3cn}{16} + \frac{9cn}{16} + \frac{cn}{16} = cn$$

\vdots

$$i \quad cn$$

سیده سعر تعداد بُر

$$1 \quad 1$$

$$2 \quad 2$$

$$4$$

$$2^{i-1}$$

$$n$$

$$n/4$$

$$n/16$$

$$n/4^{i-1}$$

حالات ممکن هزینه:

فرض کنیم ارتفاع درخت برابر با

ارتفاع راست ترین زیرگاه باشد

هزینه بُر تعداد بُر

سلسله

هزینه بُر تعداد بُر

$$\text{هزینه سعر سایز بُر} = 1 = \frac{n}{4^{i-1}} \rightarrow i = \log_4 n + 1$$

$$\text{هزینه سعر} = \text{هزینه بُرها در سطر آخر} + \text{هزینه سعرها بجز آخر} = cn \cdot \log_4 n + (2^{i-1} \times C')$$

$$= cn \log_4 n + C' \cdot 2^{\log_4 n + 1 - 1} = cn \log_4 n + 2^{\log_4 n} = cn \log_4 n + n^{\log_4 2} = cn \log_4 n + \sqrt{n}$$

$\in \Theta(n \log n)$

اگر $(3/4)^{i-1} n$ است

$$1 = \frac{n}{(4/3)^{i-1}} \rightarrow i = \log_{4/3} n + 1$$

$$\text{هزینه سعر} = cn \cdot \log_{4/3} n + C' \times 2^{\log_{4/3} n} = cn \log_{4/3} n + C' n^{\log_{4/3} 2}$$

حالات ممکن:

فرض کنیم ارتفاع درخت با

ارتفاع چوب راستین زیرگاه

برابر باشد

$$\log_{4/3}^2 \simeq 2.43 = 1 + \varepsilon_0 \quad (\varepsilon_0 = 1.43)$$

مقدار $n^{1+\varepsilon_0}$: طرفین را بر n تقسیم کنیم
 $n \log n \in O(n^{1+\varepsilon_0})$ دع مبنی بر $n^{1+\varepsilon_0}$ کمتر از $\log n$ نیست سپه :

نتیجه: $T(n) \leq \Theta(n^{\log_{4/3}^2})$

$$\Theta(n \log n) < T(n) \leq \Theta(n^{\log_{4/3}^2})$$

• همان شبکه را بر n بار زد حد پایین آن بینداشت (انتها)

$T(n) \geq \Theta(n \log n)$ این شبکه را بر n بینداشت (انتها)

$$\exists C > 0, n_0 > 0 \mid \forall n \geq n_0 \quad T(n) \geq C n \log n$$

: حکم

حالات پایه:

$$\text{if } n=2 \rightarrow T(2) \leq C \cdot 2 \lg 2 = 2C$$

$$T(2) = T(2/4) + T(3 \times 2/4) + \Theta(2) = C'' \rightarrow \text{برای عدد ثابت } C'' \text{ است باقی از ناحلوم } \rightarrow \text{برای } n=2$$

$\underbrace{\approx 0}_{\approx 1}$

$$\text{if } C'' < 2C \rightarrow \frac{C''}{2} < C \quad \text{اگر } C \text{ را بزرگتر از } \frac{C''}{2} \text{ انتخاب کنیم حالت پایه برقرار است.}$$

حالات لذار:

$$\left. \begin{array}{l} T(n/4) \leq C n/4 (\log_2 n/4) \\ T(3n/4) \leq 3 C n/4 (\log_2 3n/4) \end{array} \right\} : \text{باشد شبکه کم}$$

$$T(n) = T(n/4) + T(3n/4) + \Theta(n) \leq C \cdot n/4 (\lg_2^{n/4}) + C \cdot 3n/4 (\lg_2^{3n/4}) + \Theta(n)$$

$$\leq Cn/4 (\lg n - \lg 4) + C \cdot 3n/4 (\lg 3 + \lg n - \lg 4) + \Theta(n)$$

$$\leq \underbrace{Cn \lg n}_{\frac{Cn}{4}} - \underbrace{Cn \lg 4}_{\frac{Cn}{4}} + \underbrace{C \cdot 3n \lg 3}_{\frac{C \cdot 3n}{4}} + \underbrace{C \cdot 3n \lg n}_{\frac{C \cdot 3n}{4}} - \underbrace{C \cdot 3n \lg 4}_{\frac{C \cdot 3n}{4}} + \Theta(n)$$

$$\leq \underbrace{Cn \lg n}_{\frac{Cn}{4}} - \underbrace{Cn \lg 4}_{\frac{Cn}{4}} + C \cdot 3n/4 \lg 3 + C'' n$$

برای این سه حکم برقرار باشد : $T(n) \leq Cn \lg n \xrightarrow{\text{معنی}} -Cn \lg 4 + C \cdot 3n/4 \lg 3 + C''' n \leq 0$

$$n(-2C + 3/4 C \lg 3 + C'') \leq 0 \rightarrow \frac{C'''}{2 - 3/4 \lg 3} \leq C$$

اگر C ای انتخاب کنیم که نهادت بالا دری آن برقرار باشد روابط صورت حمل ثابت نداشت و C های زیادی وجود دارد
 $T(n) \in \Theta(n \lg n)$ در نتیجه این نهادت است و حمل درست است و

ضریب آن $\Theta(n)$ است که باز توجه به رابطه مسئله تعیین می شود.

Master theorem / قواعد

$$T(n) = aT(n/b) + f(n)$$

زیر مسئله ها با این سایر بحث را نه باشند.

$$a > 1 \quad b > 1$$

(الف) if $\exists \varepsilon > 0 \mid f(n) \in O(n^{\log_b^a - \varepsilon}) \rightarrow T(n) \in \Theta(n^{\log_b^a})$

(ب) if $f(n) \in \Theta(n^{\log_b^a}) \rightarrow T(n) \in \Theta(n^{\log_b^a} \cdot \log n)$

(ج) if $\exists \varepsilon > 0 \wedge 0 < c < 1 \mid f(n) \in \Omega(n^{\log_b^a + \varepsilon}) \wedge a f(\frac{n}{b}) \leq c \cdot f(n)$
 $\rightarrow T(n) \in \Theta(f(n))$

$n^{\log_b^a}$ $\leftarrow \textcircled{1} < \textcircled{2}$	$f(n)$? $n^{\log_b^a}$ $\textcircled{1}$ $\textcircled{2}$	شرط صافی بودن حالات
$\textcircled{2} \leq \textcircled{1} \times \log n$ $\leftarrow \textcircled{1} = \textcircled{2}$		
$f(n)$ $\leftarrow \textcircled{1} > \textcircled{2}$		

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

: حل

$$f(n) \in \Theta(n), \quad a=4, \quad b=2$$

برای الف : $\Theta(n) \leq n^{\log_2 4 - \varepsilon} \rightarrow cn \leq n^{2-\varepsilon} \rightarrow 1 \leq 2^{-\varepsilon} \rightarrow \varepsilon \leq 1$

حتماً $\varepsilon \leq 1$ باشد لطفاً درس انت بقدرات . ذهنی هم میتواند است

$$\text{حل}: T(n) = 2T(n/2) + n^2$$

$$f(n) \in \Theta(n^2) \quad ? \quad n^{\log_2^2} = n \rightarrow C \cdot n^2 \geq n^{1+\epsilon} \rightarrow \epsilon > 1$$

$$\text{شرط صناعي بدل}: a f(n/b) \leq C \cdot f(n) \quad 0 < C < 1$$

$$2f(n/2) = 2 \cdot n^2/4 = n^2/2 \leq Cn^2 \rightarrow 1/2 \leq C < 1$$

هذا يعطى حالت بديهية $\leftarrow T(n) \in \Theta(n^2)$

$$\text{حل}: T(n) = 3T(n/3) + \Theta(n)$$

$$f(n) \in \Theta(n) \quad ? \quad n^{\log_3^3} = n \rightarrow C \cdot n = n \rightarrow C=1 \rightarrow \text{حالات بديهية}$$

$$T(n) \in \Theta(n \log n)$$

$$\text{حل}: T(n) = 2T(n/2) + n \log n$$

$$f(n) \in \Theta(n \log n) \quad ? \quad n^{\log_2^2} = n \rightarrow Cn \log n > n^{1+\epsilon} \xrightarrow{\div n} \log n > n^\epsilon$$

هذه رسمة تبيّن لاريتي لزهدتاج حين جمله اي كسر است هم 4 ممبيت نعم ويرد به درجت بالاصدار لكن بحسب حالت
حالات الـ n رب هم n لا يطأها بديهية سبب حون رسم $n \log n$ از n سبب است. بحسب این مدل
حالة بديهية سبب. حالات الـ n رب هم n لا يطأها بديهية سبب حون رسم $n \log n$ از n سبب است. بحسب این مدل
نعمات باقتصنه اصلی تحلیل کرد.

extended master theorem / مصْنَعِي اسْتَمْرِثِر

وَهُنَّ حَالَتَ بِقَعْدَيْنِ اَخْرَى اَنْتَ مَدِيرَ:

$$\text{if } f(n) \in \Theta(n^{\log_b^k} + (\log n)^k) \rightarrow T(n) \in (n^{\log_b^k} \cdot (\log n)^{k+1})$$

$$f(n) = n \log n \quad ? \quad n^{\log_2^2} + (\log n)^k \rightarrow \text{equal if } k=1$$

$$\rightarrow T(n) \in \Theta(n \cdot (\log n)^2)$$

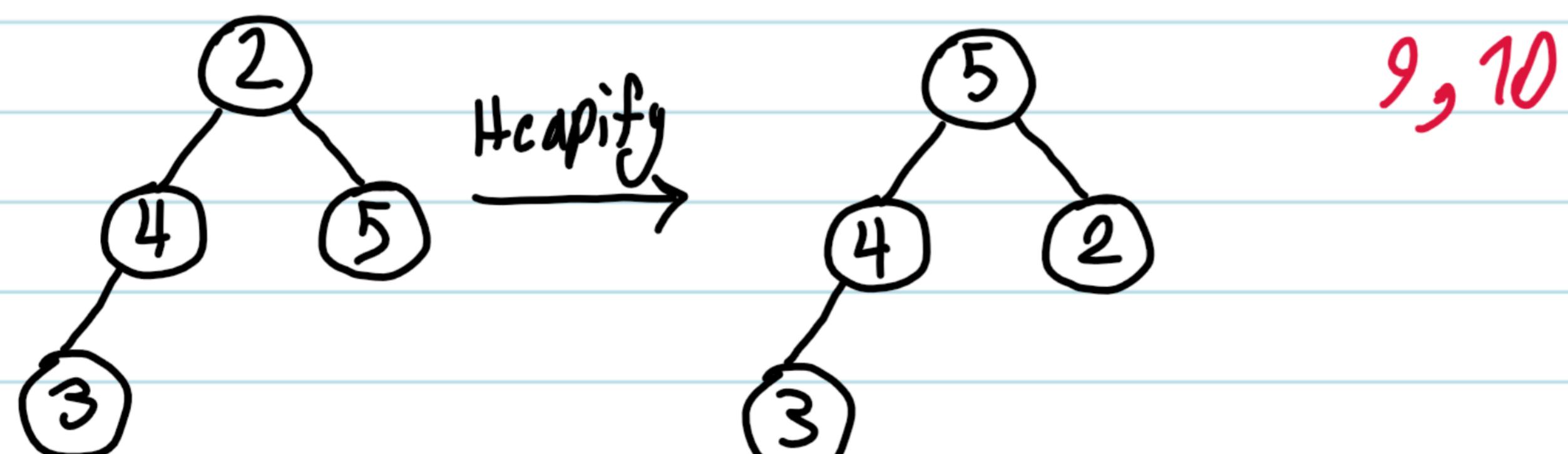
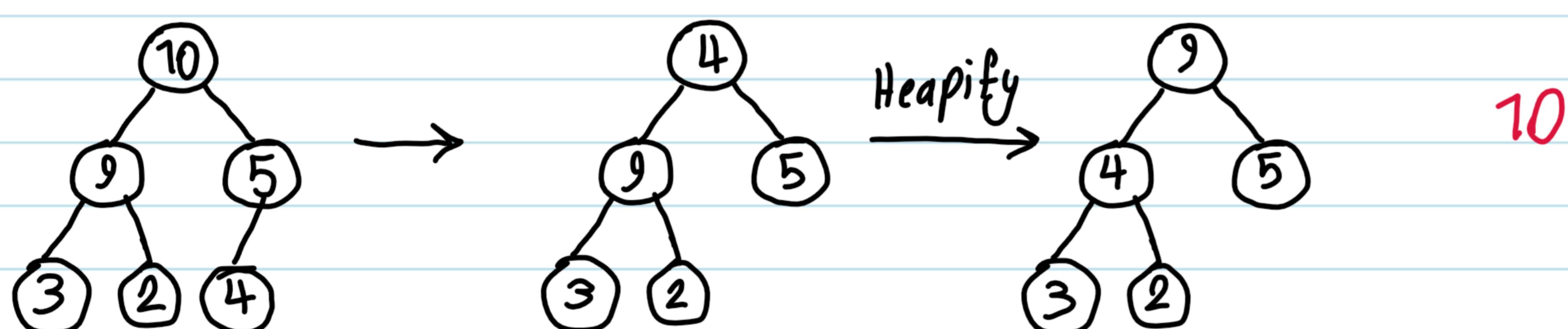
Heap sort / quick sort مِنْهُمْ مِنْهُمْ

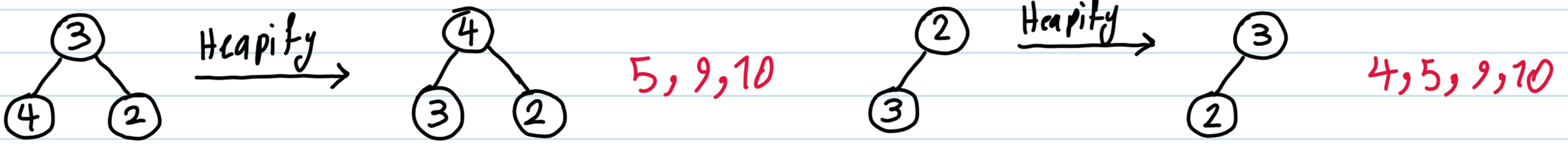
زَمَانِي وَابْلَاتِ رَسَّي : Heap sort

بَيْدَرْضَتِ بَايْنِي هَمْ سَرَّهَا بِجَزْءَهَا اَخْرَى مَالَيْرَانِ وَرَرَسَطِ اَخْرَى دَهْهَالِزِصِي بَرَسَتِ بِرَمَدِهِ : Heap

بَيْنِي دَرَسَتِ بَايْنِي هَمْ سَرَّهَا بِجَزْءَهَا اَخْرَى مَالَيْرَانِ وَرَرَسَطِ اَخْرَى دَهْهَالِزِصِي بَرَسَتِ بِرَمَدِهِ : Max/min Heap
وَرَسَيْ قَدَرَ دَارَ.

بَالْسَعَادِيَارِ : Heap sort





② $3, 4, 5, 9, 10 \rightarrow 2, 3, 4, 5, 9, 10$

Heap-Sort (A, n)

1. build-max-heap(A, n)
2. for $i = n$ to 2 :
3. swap ($A[1], A[i]$)
4. heap-size --
5. max-heapify (A, i)

: سبک

هر بار عصر آخوند آرایه (ریشه) را با اولین جایگاهی کنیم و دیگر باعث آخوند کار نداریم
برای heap-size $\leftarrow n=2$ برای 1 همیزی heap-size
پس از آخوند عصر باقی مانده را آرایه (جایگزین) و آرایه به طور صورتی
مرتب می‌فرمود

Max-Heapify (A, i)

1. $l = 2i$ $r = 2i + 1$
2. if $l \leq \text{heap-size}$ and $A[l] > A[r]$:
3. largest = l
4. else :
5. largest = i
6. if $r \leq \text{heap-size}$ and $A[r] > A[\text{largest}]$:
7. largest = r
8. if $\text{largest} \neq i$
9. swap ($A[i], A[\text{largest}]$)
10. max-heapify ($A[i], A[\text{largest}]$)

: حفظ زمانی

خط ۱ : هزینه ثابت

خط ۲ : رابطه بازگشتی