Master theorem + درخت بازگشت

طراحی الگوریتم‌ها – جلسه ششم

Introduction to Algorithm

استاد: جوانمردی

۱۳۹۹/۷/۱۹

مرور جلسه قبل

روش تقسیم و حل

The divide-and-conquer approach:

Divide the problem into a number of subproblems

Conquer the subproblems by solving them recursively

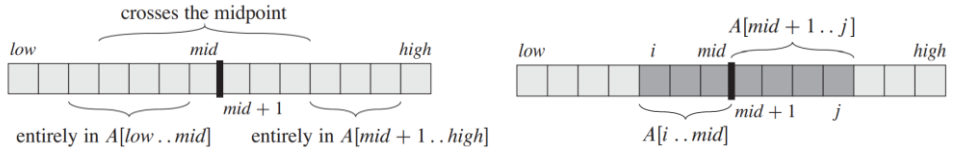
Combine subproblems and solve the original problem

مسئله زیرآرایه بهینه

- Input: A sequence $A[1], A[2], \ldots, A[n]$ of integers.
- Output: Two indicex i and j with $1 \le i \le j \le n$ that maximize

$$A[i] + A[i+1] + \cdots + A[j].$$

روش تقسیم و حل MSA




```
FIND-MAXIMUM-SUBARRAY(A, low, high)
1  if high == low
2      return (low, high, A[low])          // base case: only one element      O(1)
3  else mid = ⌊(low + high)/2⌋
Divide  (left-low, left-high, left-sum) =
        FIND-MAXIMUM-SUBARRAY(A, low, mid)   Conquer                            T(mid − low)
5       (right-low, right-high, right-sum) =
        FIND-MAXIMUM-SUBARRAY(A, mid + 1, high)                                 T(high − (mid + 1))
6       (cross-low, cross-high, cross-sum) =
        FIND-MAX-CROSSING-SUBARRAY(A, low, mid, high)                           O(high − mid)
7       if left-sum ≥ right-sum and left-sum ≥ cross-sum                        O(1)
8           return (left-low, left-high, left-sum)
9       elseif right-sum ≥ left-sum and right-sum ≥ cross-sum                   O(1)
10          return (right-low, right-high, right-sum)                  Combine
11      else return (cross-low, cross-high, cross-sum)                         O(1)
```

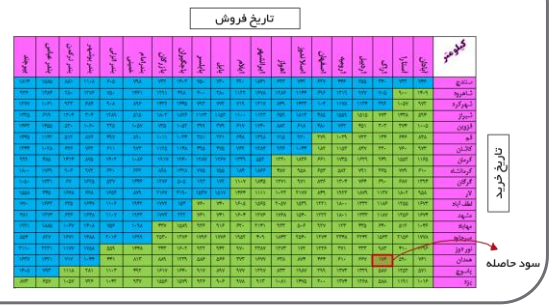
حل روابط بازگشت با جایگذاری و استقراء

- Example:
  $$T(n) = 2T(n/2) + cn$$
- Guess:
  $$T(n) = O(n \lg n)$$

⇒ Prove by induction:
$$T(n) \le dn \lg n$$
for suitable $d>0$.

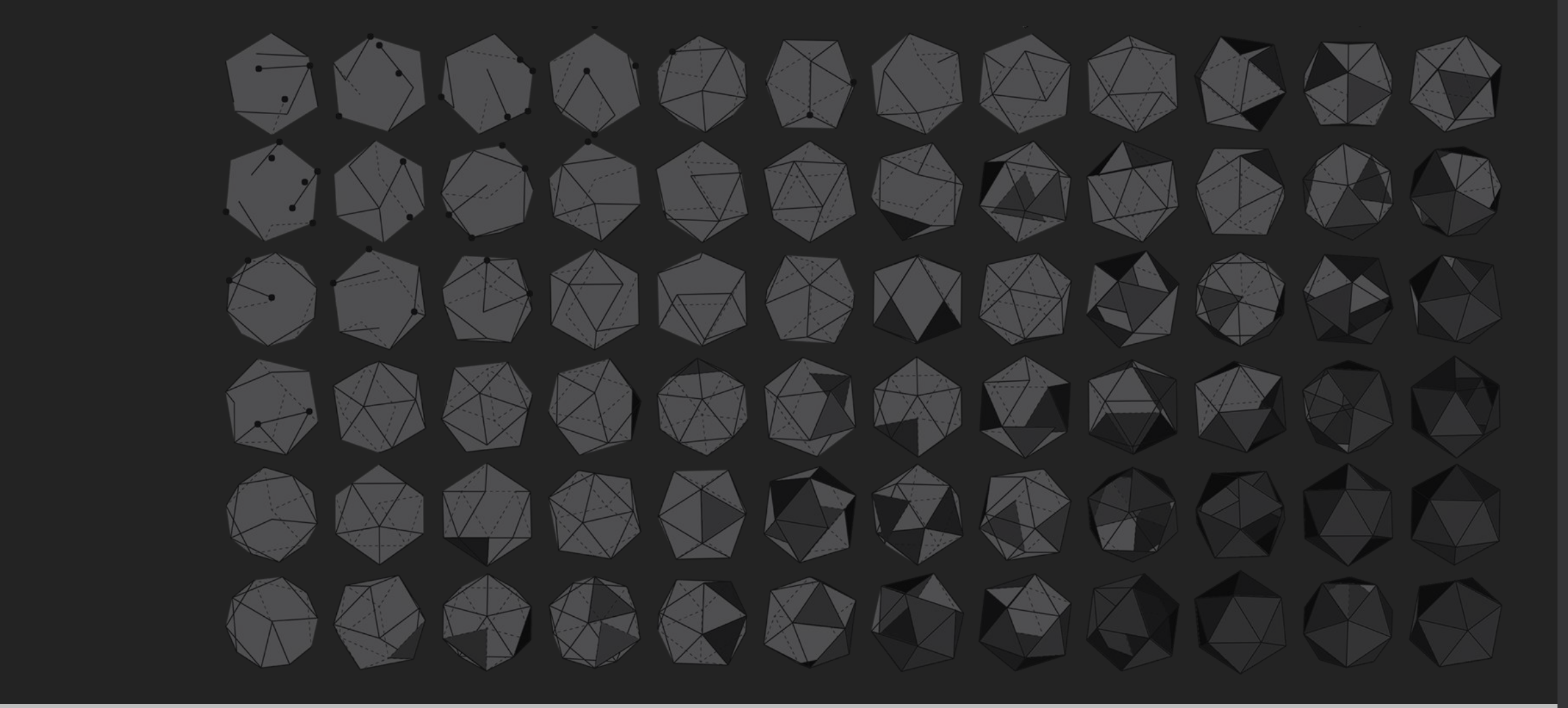
روش تهاجمی $O(n^3)$ و $O(n^2)$

`R[n]` is the sum over `A[1...n]`



فصل چهارم: تقسیم و حل | Divide and Conquer

- مسئله زیرآرایه بیشینه یا Maximum Subarray
- انواع روش‌های حل معادله بازگشتی
  - جایگذاری و استقرای ریاضی
  - درخت بازگشت
  - قضیه اصلی
- ~~الگوریتم استراسن برای ضرب ماتریسی~~



روش‌های حل روابط بازگشتی: جایگذاری

فصل ۴.۳ کتاب

چگونه حدس خوبی بزنیم؟

- روش عمومی برای یک حدس خوب وجود ندارد!
- تجربه! خلاقیت! شهود! و شاید هم درخت بازگشتی!
- اگر مشابه رابطه بازگشت را قبلا دیده اید، راه حل مشابه میتواند منطقی باشد!

$$T(n) = 2T(\lfloor n/2 \rfloor + 17) + n$$

برای n های بزرگ $\lfloor n/2 \rfloor + 17$ و $\lfloor n/2 \rfloor$ تفاوت چندانی نخواهد داشت!

- در برخی موارد مرتبه را صحیح حدس میزنیم ولی استقراء بنوعی درست کار نمیکند!!

$$T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 1$$

$$T(n) = O(n) \longrightarrow T(n) \leq cn$$

$$\begin{aligned} T(n) &\leq c\lfloor n/2 \rfloor + c\lceil n/2 \rceil + 1 \\ &= cn + 1 \text{ , } \times \end{aligned}$$

تصحیح حدس زده شده

- در برخی موارد مرتبه را صحیح حدس میزنیم ولی استقراء بنوعی درست کار نمیکند!!

$$T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 1$$

فرض $T(n) = O(n) \longrightarrow T(n) \le cn$ ✗ $\longrightarrow$ $T(n) \le cn - d$, where $d \ge 0$ is a constant.

$$\begin{aligned} T(n) &\le (c\lfloor n/2 \rfloor - d) + (c\lceil n/2 \rceil - d) + 1 \\ &= cn - 2d + 1 \\ &\le cn - d\ , \end{aligned}$$

$d \ge 1$

۱. درجه جمله اضافه کمتر از حکم باشد: به حکم یک جمله از درجه کمتر اضافه میکنیم

۲. درجه جمله اضافه با حکم برابر باشد: یک فاکتور لگاریتم در حکم کمتر حدس زدیم

۳. درجه جمله اضافه بیشتر از حکم باشد: باید حکم از درجه بالاتری باشد

تصحیح حدس زده شده

۱. درجه جمله اضافه کمتر از حکم باشد: به حکم یک جمله از درجه کمتر اضافه میکنیم

۲. درجه جمله اضافه با حکم برابر باشد: یک فاکتور لگاریتم در حکم کمتر حدس زدیم

۳. درجه جمله اضافه بیشتر از حکم باشد: باید حکم از درجه بالاتری باشد

$$T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + n$$

فرض $T(n) = O(n) \longrightarrow T(n) \leq cn$

$$\begin{aligned} T(n) &\leq c\lfloor n/2 \rfloor + c\lceil n/2 \rceil + n \\ &= cn + n, \end{aligned}$$

حالت دوم، درجه جمله اضافی برابر حکم است. پس یک لگاریتم کم حدس زده ایم!

خطاهای احتمالی در استقراء

- خطا کردن در استقراء خیلی هم سخت نیست!

$$T(n) = 2T(\lfloor n/2 \rfloor) + n$$

فرض $T(n) = O(n) \longrightarrow T(n) \leq cn$ ✗

$$\begin{aligned} T(n) &\leq 2(c\lfloor n/2 \rfloor) + n \\ &\leq cn + n \\ &= O(n), \qquad \Longleftarrow \textit{wrong!!} \end{aligned}$$

تغییر متغیر برای حل استقراء

- در برخی موارد یک تغییر متغیر ساده شما را به یک رابطه بازگشتی آشنا میرساند

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

$$m = \lg n$$

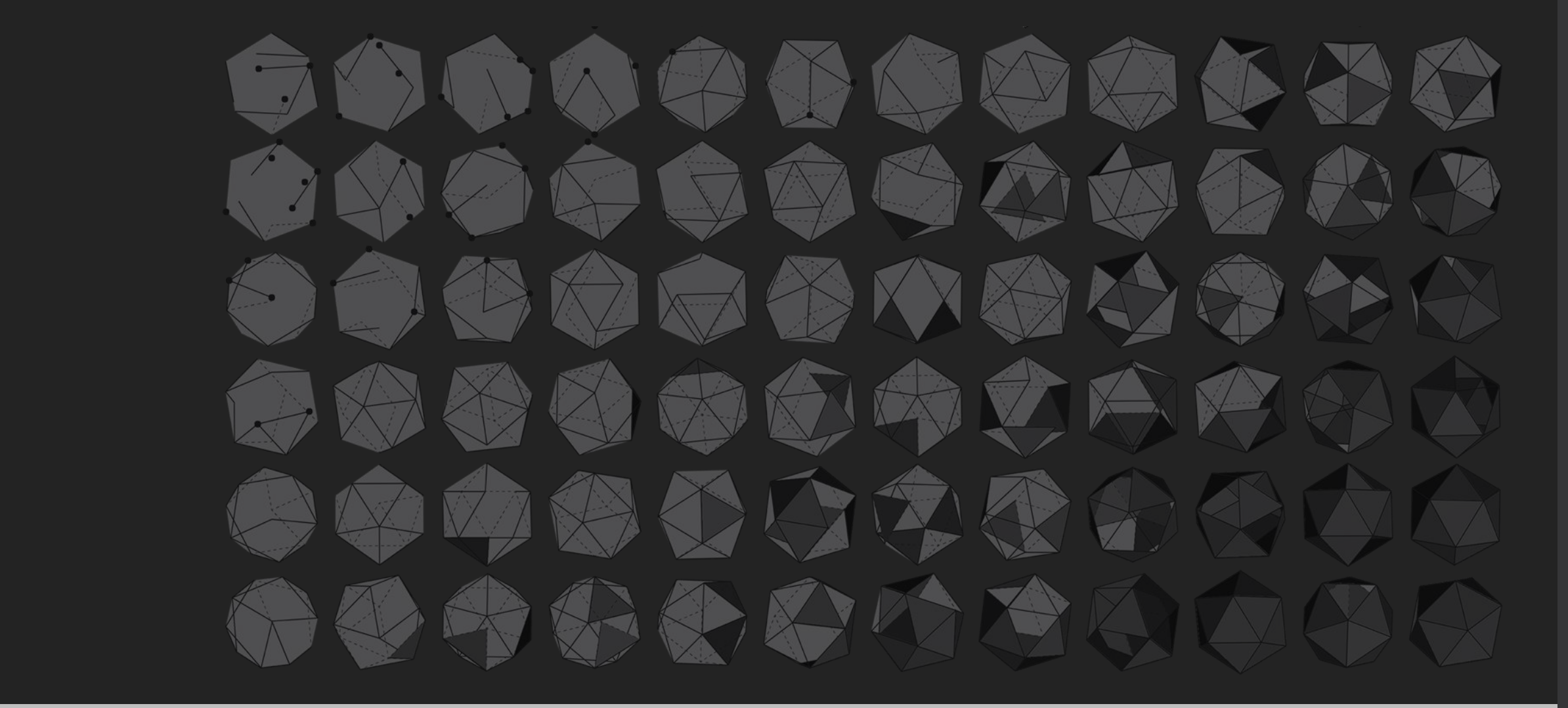
$$T(2^m) = 2T(2^{m/2}) + m$$

$$S(m) = T(2^m)$$

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

$$S(m) = O(m \lg m)$$

$$T(n) = T(2^m) = S(m) = O(m \lg m) = O(\lg n \lg \lg n)$$



روش‌های حل رابطه بازگشتی: درخت بازگشت

فصل ۴.۴ کتاب

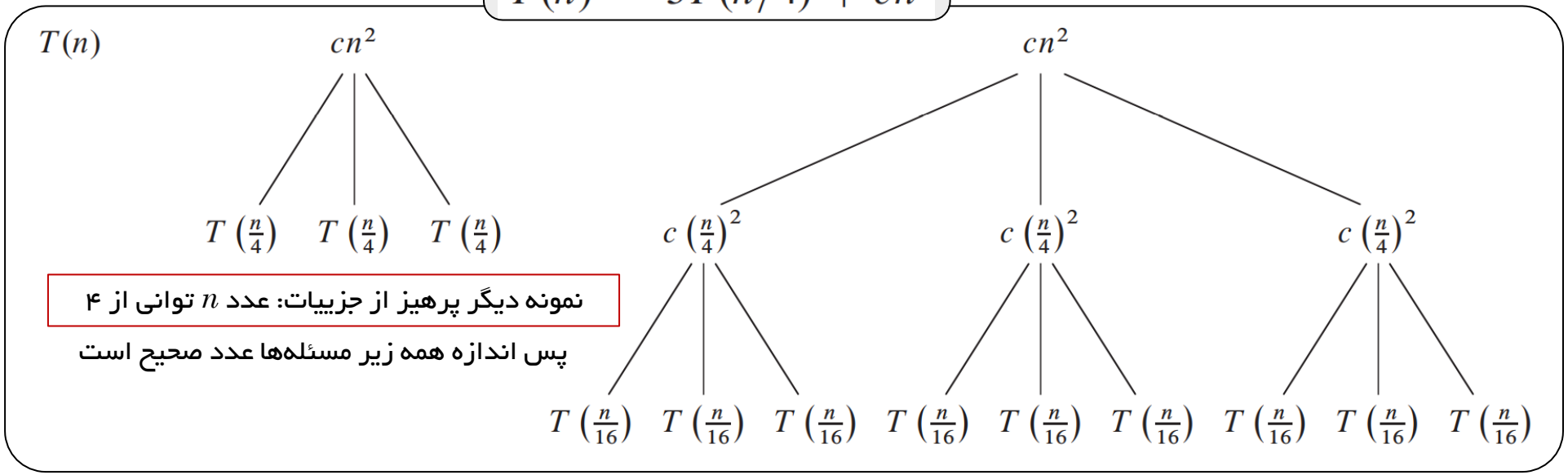
روش درخت بازگشتی

- روش جایگذاری و استقراء مناسب برای حل روابط بازگشتی
- اما پیدا کردن حدس مناسب همیشه راحت نیست! ⟵ درخت بازگشتی
- در محاسبات درخت بازگشت میتوان از برخی جزییات پرهیز کرد
- درآن صورت اثبات حدس به دست آمده از طریق استقراء ضرورت خواهد داشت

$$T(n) = 3T(\lfloor n/4 \rfloor) + \Theta(n^2)$$

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

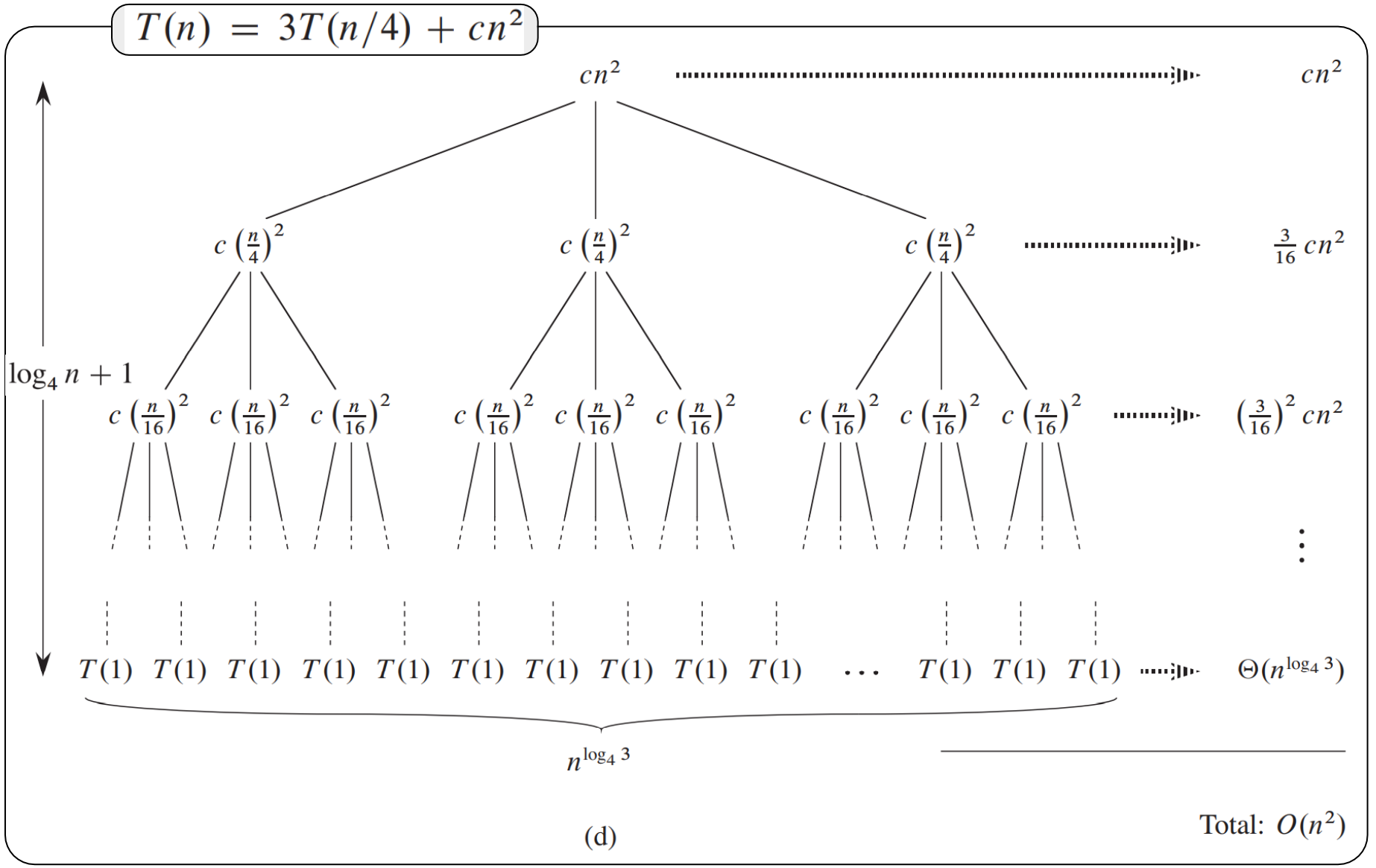
نمونه پرهیز از جزییات: محاسبه حد بالا با حذف کف



نمونه دیگر پرهیز از جزییات: عدد n توانی از ۴

پس اندازه همه زیر مسئله‌ها عدد صحیح است

روش درخت بازگشتی



محاسبه مجموع هم سطح‌ها

محاسبه تعداد سطوح

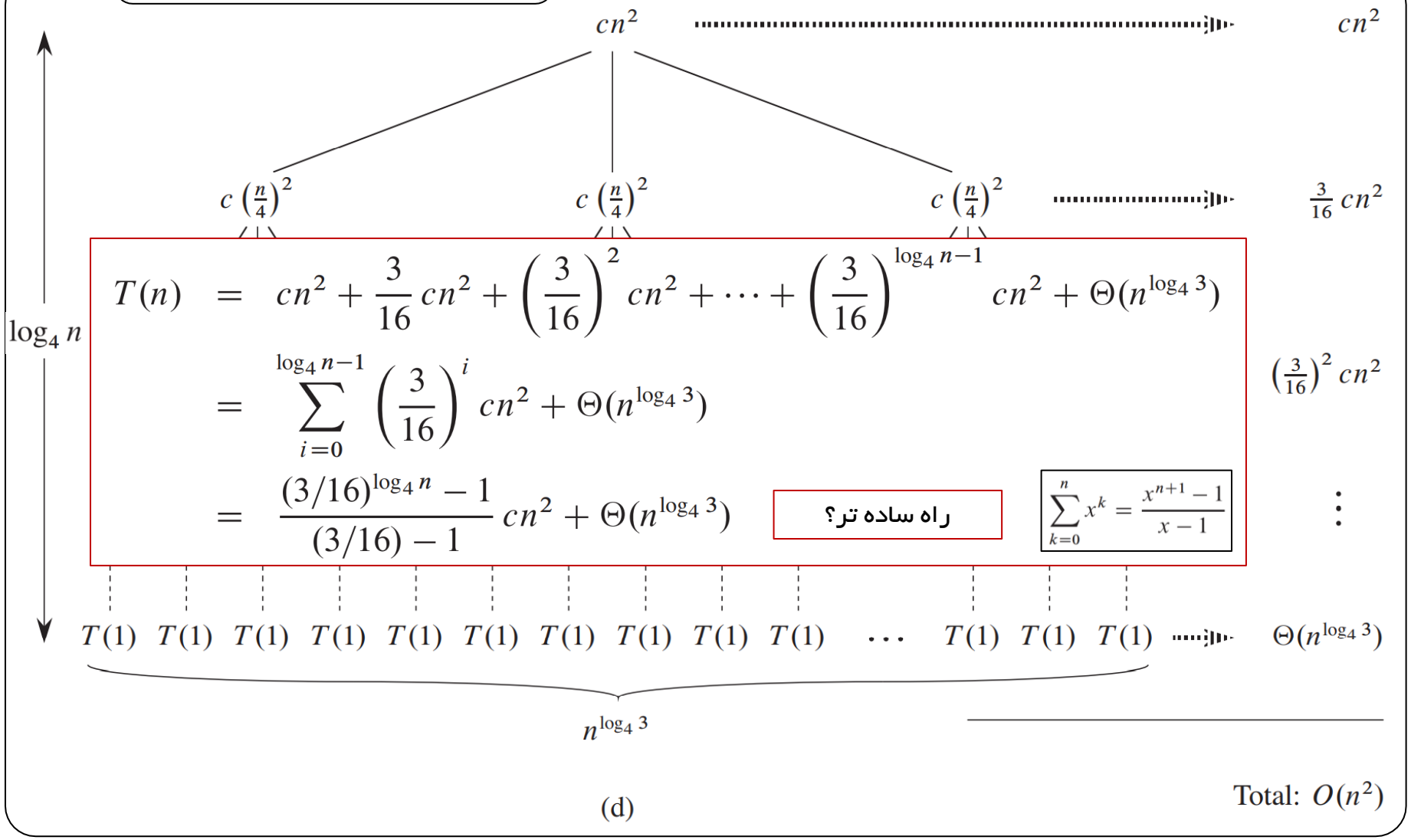
$n/4^i = 1$

محاسبه مجموع سطوح

$\log_4 3$ 0.79248125 . . .

روش درخت بازگشتی

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



$$T(n) = cn^2 + \frac{3}{16}cn^2 + \left(\frac{3}{16}\right)^2 cn^2 + \cdots + \left(\frac{3}{16}\right)^{\log_4 n - 1} cn^2 + \Theta(n^{\log_4 3})$$

$$= \sum_{i=0}^{\log_4 n - 1} \left(\frac{3}{16}\right)^i cn^2 + \Theta(n^{\log_4 3})$$

$$= \frac{(3/16)^{\log_4 n} - 1}{(3/16) - 1} cn^2 + \Theta(n^{\log_4 3})$$

راه ساده تر؟

$$\sum_{k=0}^{n} x^k = \frac{x^{n+1} - 1}{x - 1}$$

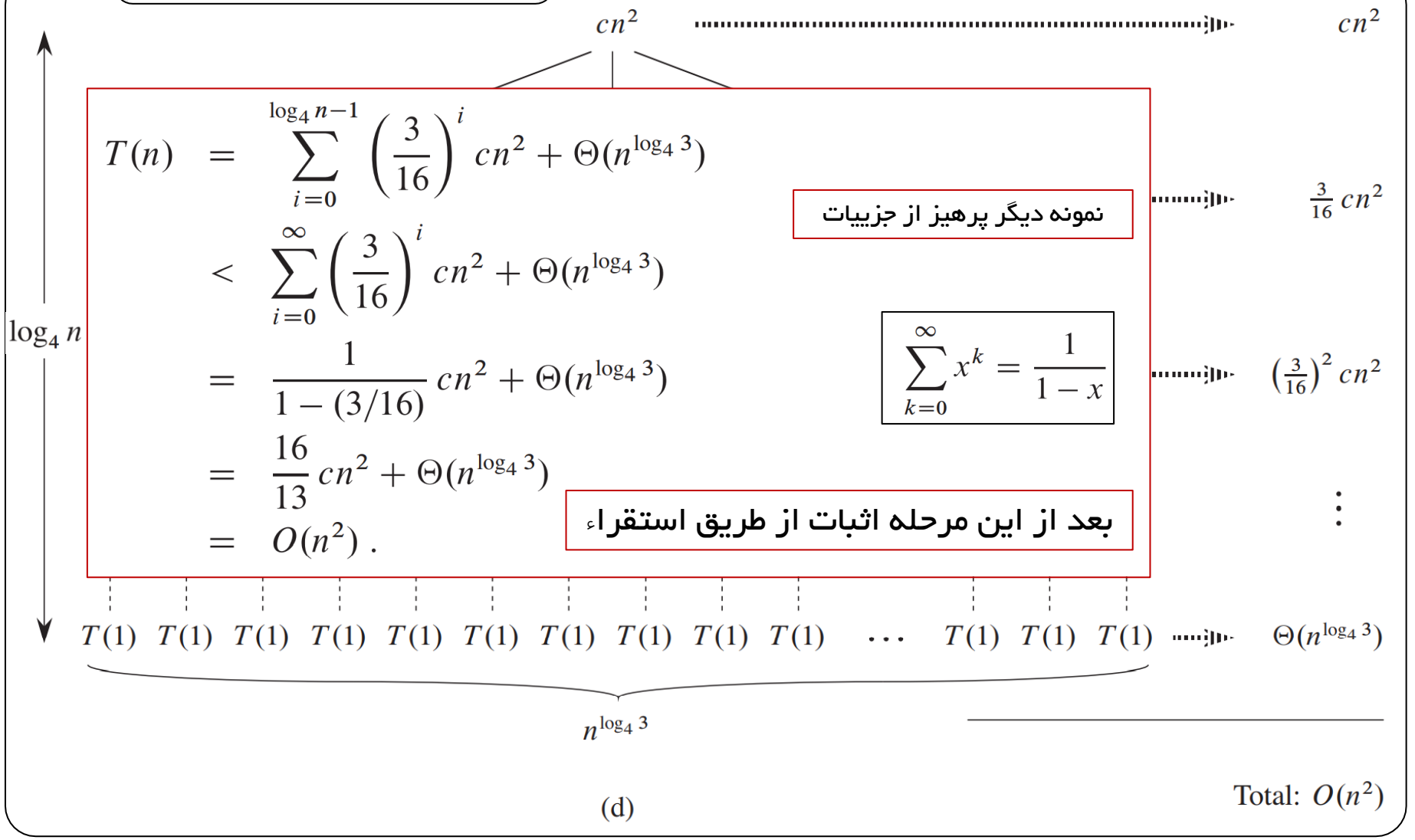
محاسبه مجموع هم سطح‌ها

محاسبه تعداد سطوح

محاسبه مجموع سطوح

روش درخت بازگشتی

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



$$\begin{aligned} T(n) &= \sum_{i=0}^{\log_4 n - 1} \left(\frac{3}{16}\right)^i cn^2 + \Theta(n^{\log_4 3}) \\ &< \sum_{i=0}^{\infty} \left(\frac{3}{16}\right)^i cn^2 + \Theta(n^{\log_4 3}) \\ &= \frac{1}{1-(3/16)} cn^2 + \Theta(n^{\log_4 3}) \\ &= \frac{16}{13} cn^2 + \Theta(n^{\log_4 3}) \\ &= O(n^2) . \end{aligned}$$

نمونه دیگر پرهیز از جزییات

$$\sum_{k=0}^{\infty} x^k = \frac{1}{1-x}$$

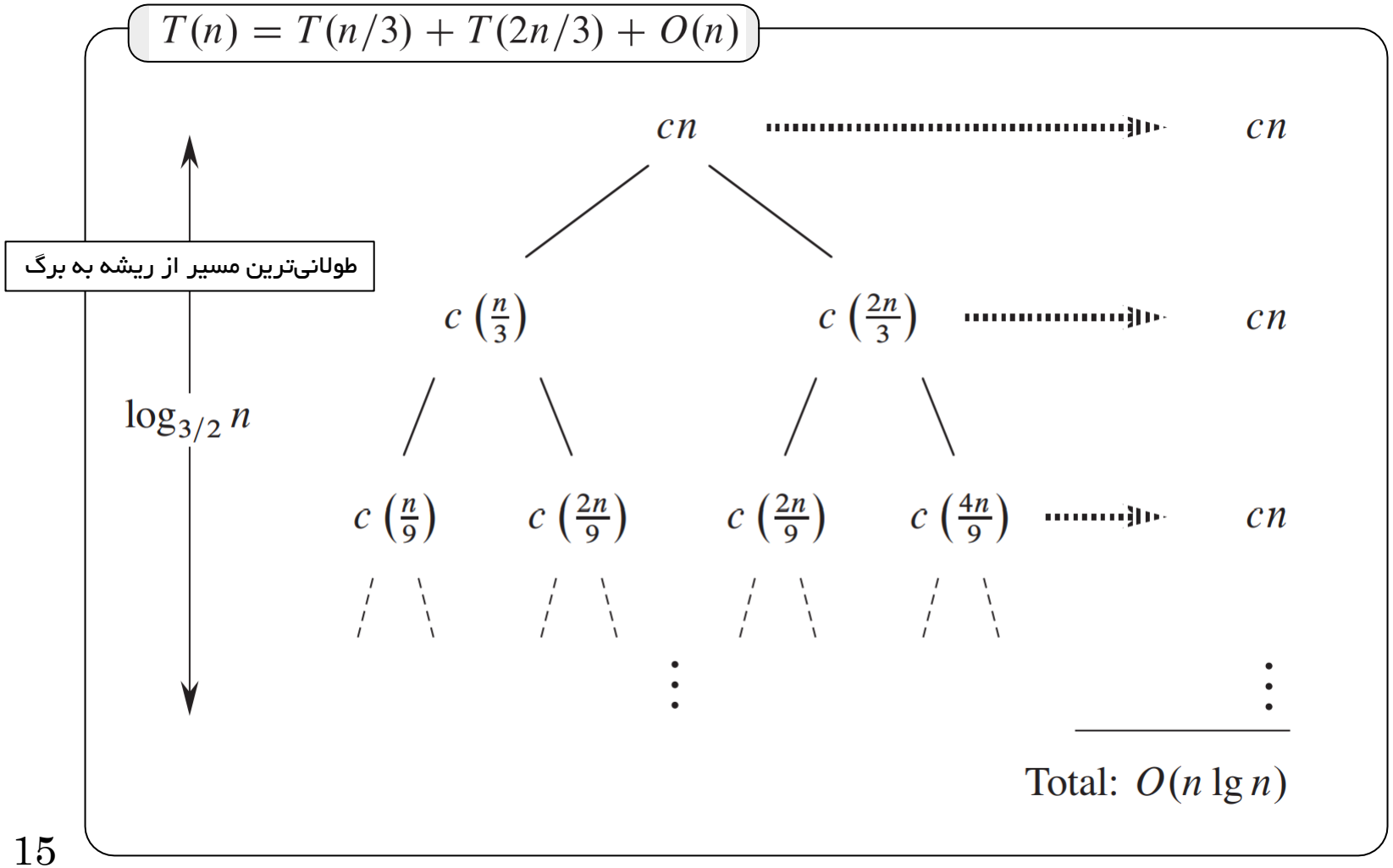
بعد از این مرحله اثبات از طریق استقراء

محاسبه مجموع هم سطح‌ها

محاسبه تعداد سطوح

محاسبه مجموع سطوح

روش درخت بازگشتی



طولانی‌ترین مسیر از ریشه به برگ

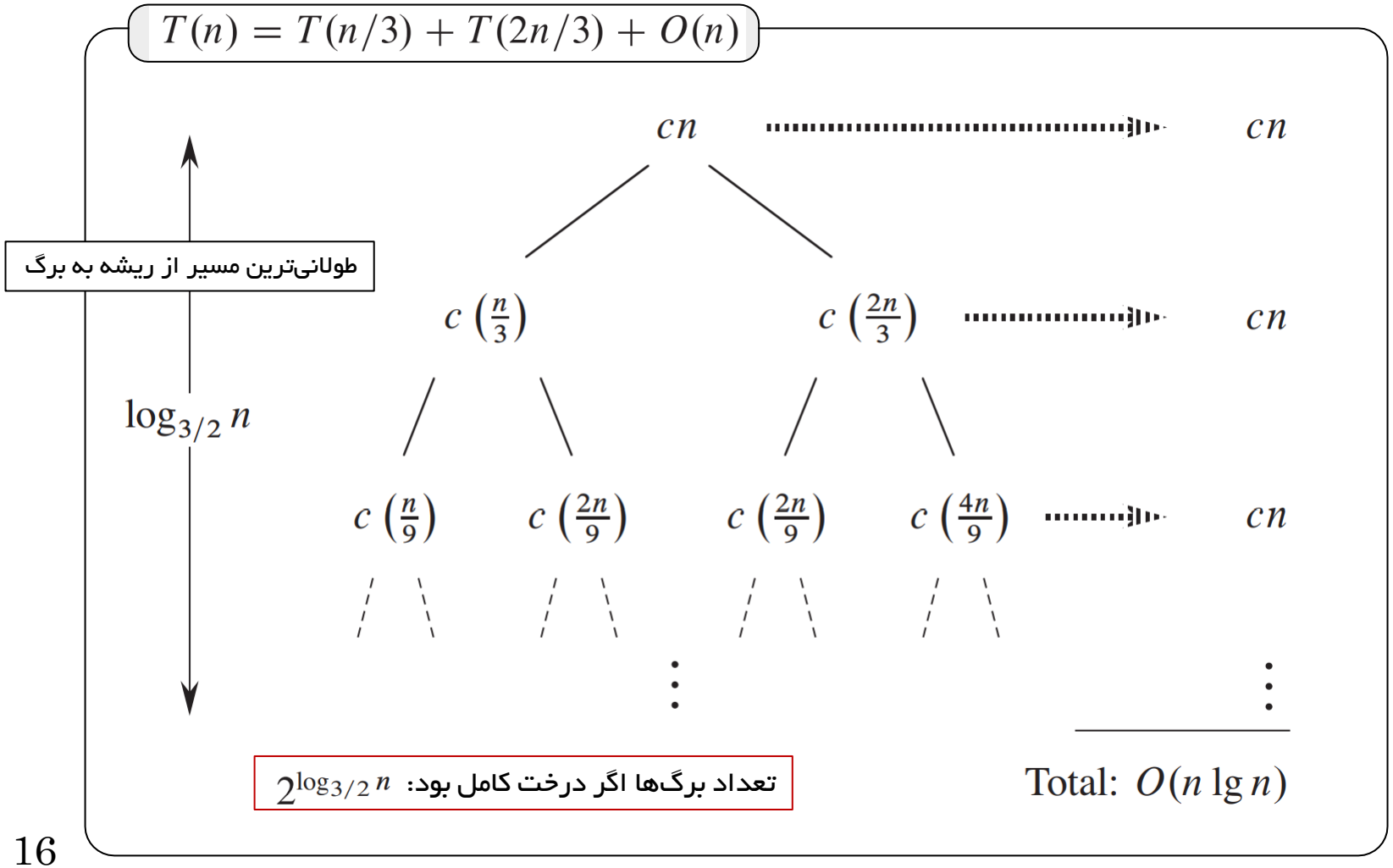
محاسبه مجموع هم سطح‌ها

محاسبه تعداد سطوح

$(2/3)^k n = 1$ when $k = \log_{3/2} n$

محاسبه مجموع سطوح

روش درخت بازگشتی



طولانی‌ترین مسیر از ریشه به برگ

تعداد برگ‌ها اگر درخت کامل بود: $2^{\log_{3/2} n}$

محاسبه مجموع هم سطح‌ها

آیا درخت کامل است؟

آیا مجموع سطوح همیشه cn است؟

همه اینها را می‌توان دقیق محاسبه کرد اما...

نمونه‌هایی از پرهیز در محاسبه جزییات

محاسبه مجموع سطوح

روش درخت بازگشتی

$$T(n) = T(n/3) + T(2n/3) + O(n)$$

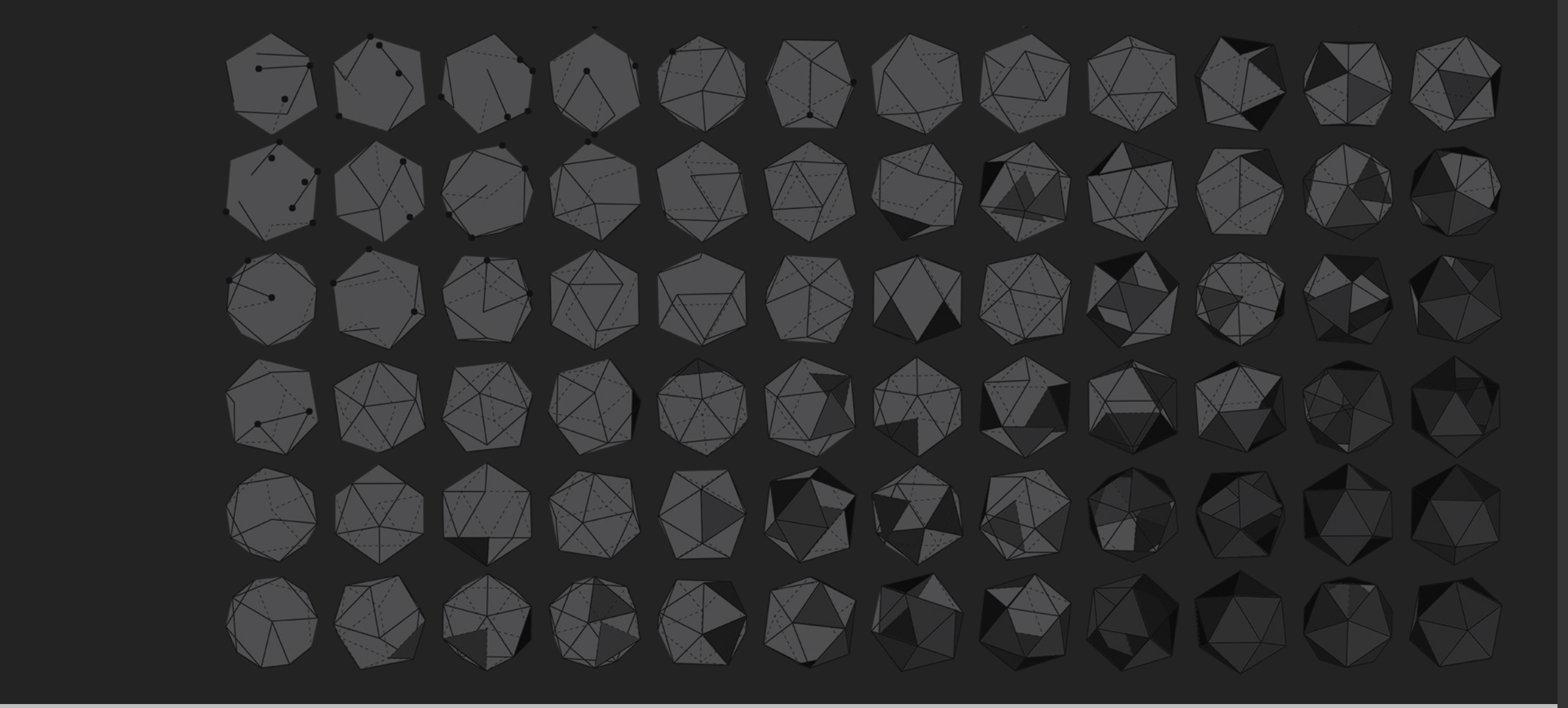
Total: $O(n \lg n)$

$$T(n) \le d n \lg n$$

$$\begin{aligned}
T(n) &\le T(n/3) + T(2n/3) + cn \\
&\le d(n/3)\lg(n/3) + d(2n/3)\lg(2n/3) + cn \\
&= (d(n/3)\lg n - d(n/3)\lg 3) \\
&\qquad + (d(2n/3)\lg n - d(2n/3)\lg(3/2)) + cn \\
&= dn\lg n - d((n/3)\lg 3 + (2n/3)\lg(3/2)) + cn \\
&= dn\lg n - d((n/3)\lg 3 + (2n/3)\lg 3 - (2n/3)\lg 2) + cn \\
&= dn\lg n - dn(\lg 3 - 2/3) + cn \\
&\le dn\lg n\ ,
\end{aligned}$$

$$d \ge c/(\lg 3 - (2/3))$$

اثبات حالت‌های پایه:
چه بازه ای برای d درنظر گرفته شود



روش‌های حل رابطه بازگشتی: قضیه اصلی Master Theorem

فصل ۴.۵ کتاب

قضیه اصلی Mater Theorem

$$T(n) = aT(n/b) + f(n) \qquad a \geq 1 \text{ and } b > 1$$

رابطه بازگشت $T(n)$ مسئله را به a زیرمسئله مساوی به اندازه n/b تقسیم میکند و هزینه تقسیم و ترکیب زیرمسئله‌ها برابر $f(n)$ است

n/b to mean either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(f(n) \lg n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$.

شرط منظم بودن

$$f(n) \square n^{\log_b a}$$

مثال قضیه اصلی – ۱

$$T(n) = 9T(n/3) + n$$

$$T(n) = aT(n/b) + f(n) \qquad a \geq 1 \text{ and } b > 1$$

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(f(n) \lg n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

$$n^{\log_b a} = n^{\log_3 9} = \Theta(n^2)$$

$$f(n) = O(n^{\log_3 9 - \epsilon}), \text{ where } \epsilon = 1$$

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

مثال قضیه اصلی – ۲

$$T(n) = T(2n/3) + 1$$

$$T(n) = aT(n/b) + f(n) \qquad a \geq 1 \text{ and } b > 1$$

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(f(n) \lg n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

$$n^{\log_b a} = n^{\log_{3/2} 1} = n^0 = 1$$

$$f(n) = \Theta(n^{\log_b a}) = \Theta(1)$$

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

مثال قضیه اصلی – ۳

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

$$T(n) = aT(n/b) + f(n) \qquad a \geq 1 \text{ and } b > 1$$

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(f(n) \lg n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

$$n^{\log_b a} = n^{\log_4 3} = O(n^{0.793})$$

$$f(n) = \Omega(n^{\log_4 3 + \epsilon}), \text{ where } \epsilon \approx 0.2$$

$$af(n/b) = 3(n/4)\lg(n/4) \leq (3/4)n \lg n = cf(n) \text{ for } c = 3/4$$

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

مثال قضیه اصلی – ۴

$$T(n) = 2T(n/2) + n \lg n$$

$n^{\log_b a} = n$ □ $f(n) = n \lg n$

is not *polynomially* larger

$$T(n) = aT(n/b) + f(n) \qquad a \geq 1 \text{ and } b > 1$$

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(f(n) \lg n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

The ratio $f(n)/n^{\log_b a} = (n \lg n)/n = \lg n$ is asymptotically less than n^ϵ for any positive constant ϵ

the recurrence falls into the gap between case 2 and case 3

Exercise 4.6-2

Show that if $f(n) = \Theta(n^{\log_b a} \lg^k n)$, where $k \geq 0$, then the master recurrence has solution $T(n) = \Theta(n^{\log_b a} \lg^{k+1} n)$. For simplicity, confine your analysis to exact powers of b.

مثال قضیه اصلی – ۵

$$T(n) = aT(n/b) + f(n) \qquad a \geq 1 \text{ and } b > 1$$

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(f(n) \lg n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

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

مرتبه زمانی الگوریتم‌های تقسیم و حل مرتب‌سازی ترکیبی و زیرآرایه بیشینه

$$n^{\log_b a} = n^{\log_2 2} = n$$

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

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

مثال قضیه اصلی – ۶

$$T(n) = aT(n/b) + f(n) \qquad a \geq 1 \text{ and } b > 1$$

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(f(n) \lg n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

$$T(n) = 2T(n/4) + \sqrt{n}$$

مثال قضیه اصلی – ۸

استفاده از تغییر متغیر

$$T(n) = T(\sqrt{n}) + 1$$

$$T(n) = aT(n/b) + f(n) \qquad a \geq 1 \text{ and } b > 1$$

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(f(n) \lg n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

$$m = \lg n$$

$$S(m) = T(2^m)$$

$$T(2^m) = T(2^{m/2}) + 1$$

$$S(m) = S(m/2) + 1$$

$$n^{\log_b a} = n^{\log_2 1} = n^0 = 1 \qquad f(n) = 1$$

$$1 = \Theta(1)$$

case 2 applies and $S(m) = \Theta(\lg m)$

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