



دانشگاه شهید بهشتی کرج

# پک: تحلیل الگوریتم ها

ساختمان داده ها و الگوریتم

مدرس: دکتر نجمه منصوری

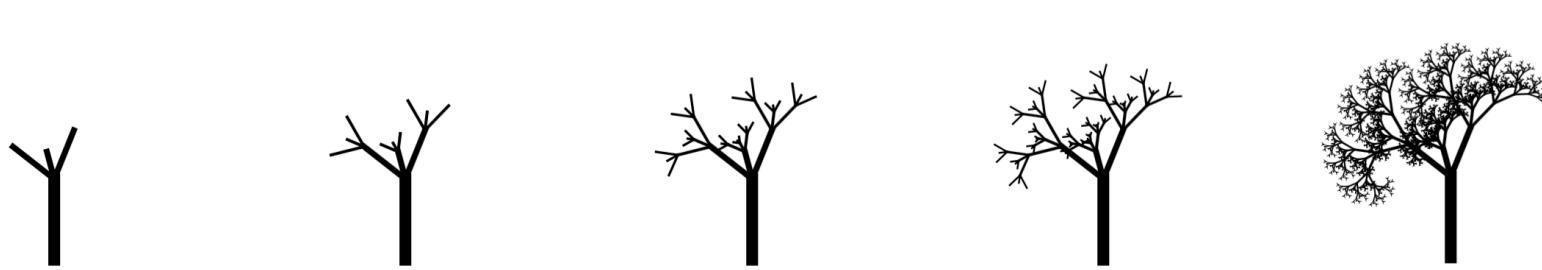
نگارنده: سجاد هاشمیان



# تابع بازگشتی

تابع بازگشتی تابعی است که خودش را به صورت مستقیم یا غیر مستقیم فراخوانی می‌کند.

- **سورس کد کوتاه:** برخی از الگوریتم‌ها را به راحتی می‌توان به صورت بازگشتی با دستورات کمتری نسبت به حالت غیر بازگشتی پیاده سازی کرد.
- **اتلاف حافظه بیشتر:** به علت استفاده از حافظه اضافی stack که مراحل بازگشت را نگهداری می‌کند، اتلاف حافظه نسبت به حالت غیر بازگشتی بیشتر است.
- **سرعت اجرا کمتر:** سرعت اجرا در بیشتر الگوریتم‌های بازگشتی کمتر از الگوریتم‌های غیربازگشتی می‌باشد.



# مجموع اعداد از ۱ تا $n$

```
def sum(n):  
    if(n==1):  
        return 1  
    else:  
        return sum(n-1)+n
```

$$sum(n) = \begin{cases} 1 & n = 1 \\ n + sum(n - 1) & n > 1 \end{cases}$$

خروجی فراخوانی تابع به ازای  $n=3$  برابر با 6 است:

```
sum(3)=3+sum(2) =3+3=6  
sum(2)=2+sum(1) =2+1=3  
sum(1)=1
```

# تابع توان $n^m$

```
def pow(n,m):  
    if(m==0):  
        return 1  
    else:  
        return pow(n,m-1)*n
```

$$pow(n, m) = \begin{cases} 1 & m = 0 \\ n \times pow(n, m - 1) & m > 0 \end{cases}$$

خروجی فراخوانی تابع  $pow(3,3)$  برابر با 27 است:

```
pow(3,3)=3*pow(3,2) =3*9=27  
pow(3,2)=3*pow(3,1) =3*3=9  
pow(3,1)=3*pow(3,0) =3*1=3  
pow(3,0)=1
```

# تابع ترکیب $\binom{n}{m}$

```
def C(n, m):  
    if(m == 0 or n==m):  
        return 1  
    else:  
        return C(n-1,m)+C(n-1,m-1)
```

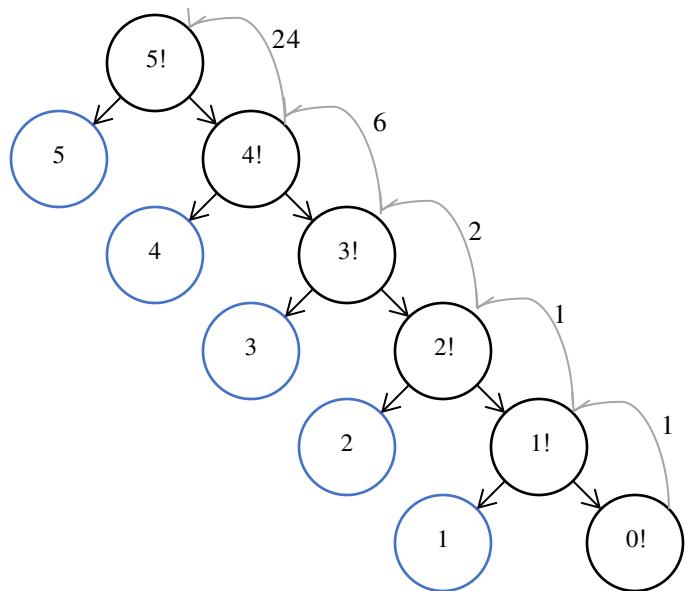
$$C(n, m) = \begin{cases} 1 & m = 0 \text{ or } m = n \\ \binom{n-1}{m} + \binom{n-1}{m-1} & 0 < m < n \end{cases}$$

خروجی فراخوانی تابع  $C(4,2)$  برابر با 6 است:

$C(4,2)=C(3,2)+C(3,1)=3+3=6$   
 $C(3,1)=C(2,1)+C(2,0)=2+1=3$   
 $C(2,0)=1$   
 $C(3,2)=C(2,2)+C(2,1)=1+2=3$   
 $C(2,1)=C(1,1)+C(1,0)=1+1=2$   
 $C(1,0)=1$   
 $C(1,1)=1$   
 $C(2,2)=1$

# تابع فاکتوریل $n!$

$$f(n) = \begin{cases} 1 & n = 0 \\ f(n - 1) \times n & n > 0 \end{cases}$$



برای فراخوانی  $f(5)$  داریم:

```
fact(5)=5*fact(4)=120
fact(4)=4*fact(3)=24
fact(3)=3*fact(2)=6
fact(2)=2*fact(1)=2
fact(1)=1*fact(0)=1
fact(0)=1
```

# محاسبه خارج قسمت

$$div(a, b) = \begin{cases} 0 & a < b \\ div(a - b, b) + 1 & a \geq b \end{cases}$$

خروجی فرآخوانی تابع  $div(11,3)$  برابر با 3 است:

```
div(11,3)=div(8,3)+1=3
div(8,3) =div(5,3)+1=2
div(5,3) =div(2,3)+1=1
div(2,3) =0
```

# بزرگترین مقسوم علیه مشترک

یافتن بزرگترین عدد صحیحی که  $a$  و  $b$  هر دو بر آن بخش پذیرند.

$$gcd(a, b) = \begin{cases} a & b = 0 \\ gcd(b, a \% b) + 1 & b \neq 0 \end{cases}$$

خروجی فراخوانی تابع  $gcd(4032, 1272)$  برابر با 24 است:

**gcd(4032, 1272) =**

=gcd(1272, 216)

=gcd(216, 192)

=gcd(192, 24)

=gcd(24, 0) = 24

4032: 1 2 2 2 2 2 2 3 3 7  
1272: 1 2 2 2 3 53

# تابع آکرمون

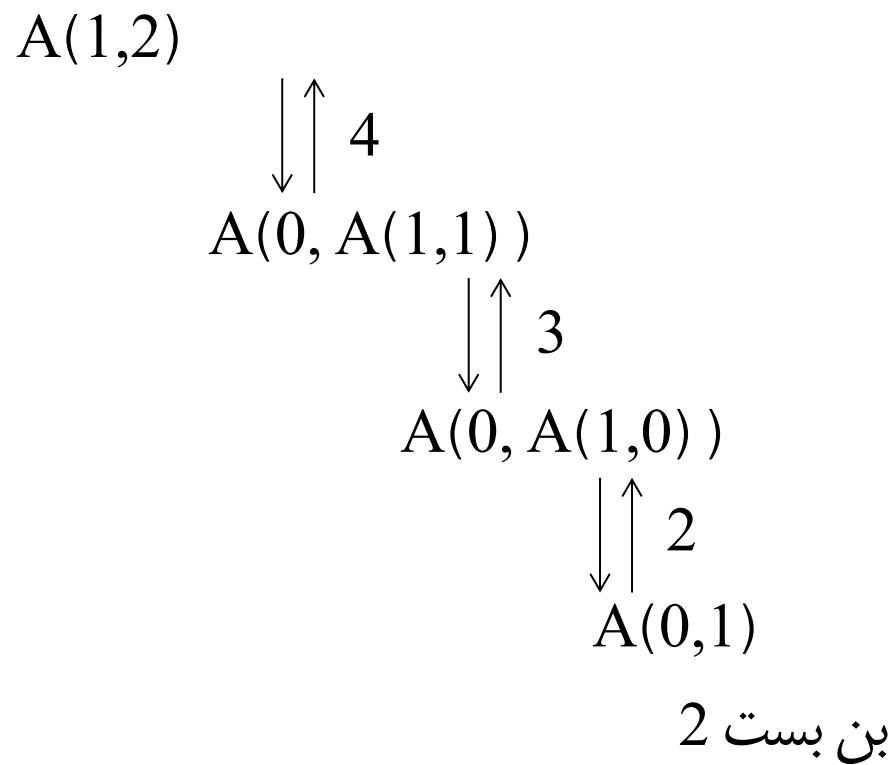
$$A(m, n) = \begin{cases} n + 1 & \text{if } m = 0 \\ A(m - 1, 1) & \text{if } m > 0 \text{ and } n = 0 \\ A(m - 1, A(m, n - 1)) & \text{if } m > 0 \text{ and } n > 0. \end{cases}$$

خروجی فرآخوانی تابع  $A(1,1)$  برابر با 3 است:

$$\begin{aligned} A(1,1) &= A(0, A(1,0)) \\ &= A(0, A(0,1)) \\ &= A(0, 2) = 3 \end{aligned}$$

$$A(m, n) = \begin{cases} n + 1 & \text{if } m = 0 \\ A(m - 1, 1) & \text{if } m > 0 \text{ and } n = 0 \\ A(m - 1, A(m, n - 1)) & \text{if } m > 0 \text{ and } n > 0. \end{cases}$$

خروجی فرآخوانی تابع  $A(1,2)$  چقدر است؟

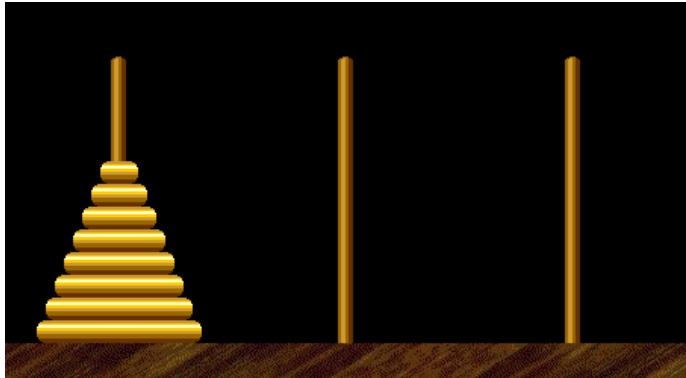


# برج های هانوی

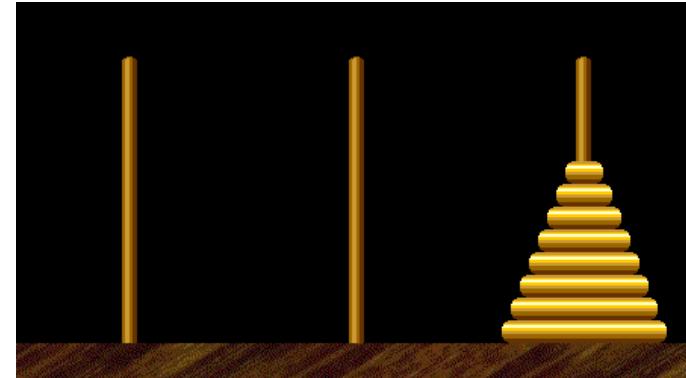
مسئله. انتقال دیسک‌ها از میله‌ی سمت چپ به میله‌ی سمت راست با حفظ ترتیب.

- در هر حرکت تنها مجاز به انتقال یک دیسک هستیم.
- هیچ گاه مجاز نیستیم یک دیسک بزرگتر را بر روی یک دیسک کوچکتر قرار دهیم.

حالت ابتدایی



حالتنهایی



# افسانه برج های هانوی

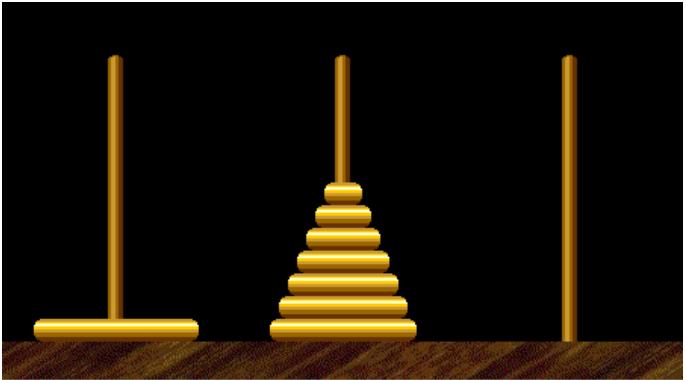


در محوطه معبدی در آسیای دور سه میله الماسی قرار داشت که یکی از آن‌ها حاوی تعدادی قرص طلایی بود. کاهنان معبد در تلاش بودند تا قرص‌های طلایی را از آن میله به یکی دیگر از میله‌ها تحت شرایطی انتقال دهند، و باور داشتند که با تمام شدن انتقال قرص‌ها عمر جهان نیز به پایان خواهد رسید!

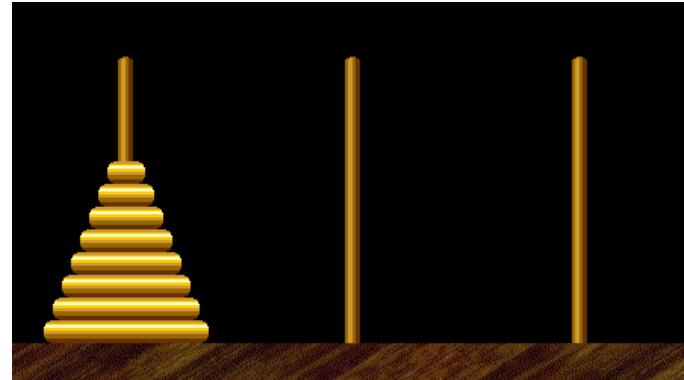
میله اولیه ۶۴ قرص داشت، که بر روی هم به‌طور نزولی بر اساس اندازه‌شان چیده شده بودند.

برگرفته از ویکی‌پدیا

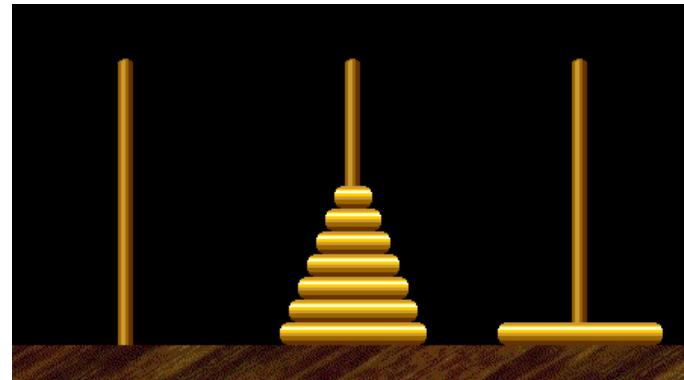
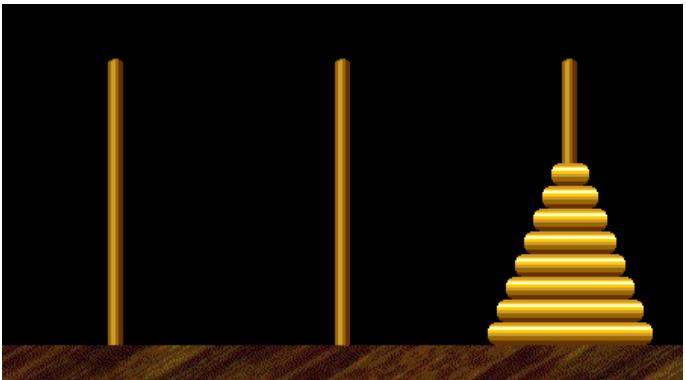
# برج های هانوی



۲. انتقال بزرگترین دیسک به میله‌ی راست



۱. انتقال  $1 - N$  دیسک به میله‌ی وسط



۳. انتقال  $1 - N$  دیسک به میله‌ی راست

# بالاخره سرنوشت دنیا چه شد؟

تعداد جابجایی‌های لازم برای انتقال  $n$  دیسک؟ برای حل یک مسئله با  $n$  دیسک به  $1 - 2^n$  جابجایی نیاز است!!! بنابراین اگر هر جابه‌جایی تنها یک ثانیه طول بکشد، دنیا ۵۸۵ میلیارد سال پس از خلقت آن به پایان میرسد. پس هنوز بیش از ۵۷۰ میلیارد سال باقیمانده است.

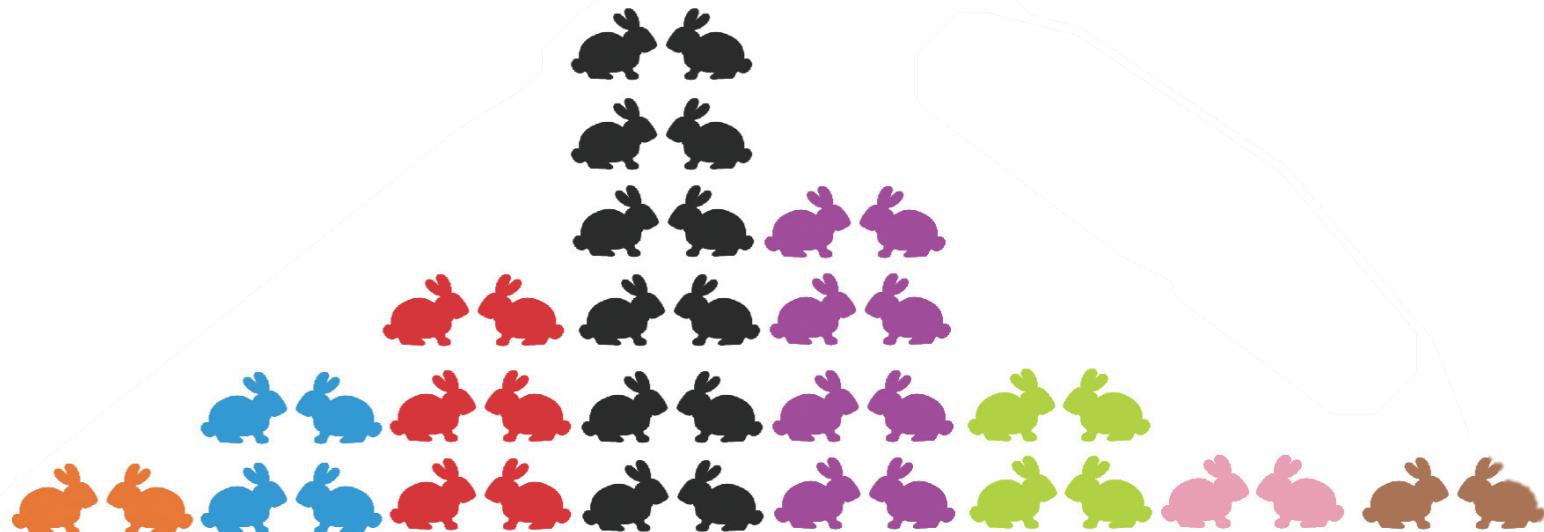
برای این که خیالتان راحت شود، این راه حل سریع‌ترین راه حل ممکن است. (واقعاً چرا؟)

```
def hanoi(nDisk, start, temp, finish):
    if(nDisk==1):
        print(f'{start} --> {finish}')
    else:
        hanoi(nDisk - 1, start, finish, temp)
        print(f'{start} --> {finish}')
        hanoi(nDisk - 1, temp, start, finish)
```

برای مثال فراخوانی تابع به شکل (hanoi(3, 'A', 'B', 'C')) مسئله برج هانوی را با سه دیسک که در میله A قرار دارند و با کمک میله B به میله C منتقل خواهد شد، حل می‌کند. نظرتان در رابطه با زمان اجرا (hanoi(64, 'A', 'B', 'C')) چیست؟

# اعداد فیبوناچی

فرض کنیم خرگوش‌هایی وجود دارند که هر جفت (یک نر و یک ماده) از آنها که به سن ۱ ماهگی رسیده باشند به ازای هر ماه که از زندگی شان سپری شود یک جفت خرگوش متولد می‌کنند که آنها هم از همین قاعده پیروی می‌کنند. حال اگر فرض کنیم این خرگوش‌ها هرگز نمی‌میرند و در آغاز یک جفت از این نوع خرگوش در اختیار داشته باشیم که به تازگی متولد شده‌اند، حساب کنید پس از  $n$  ماه چند جفت از این نوع خرگوش خواهیم داشت.



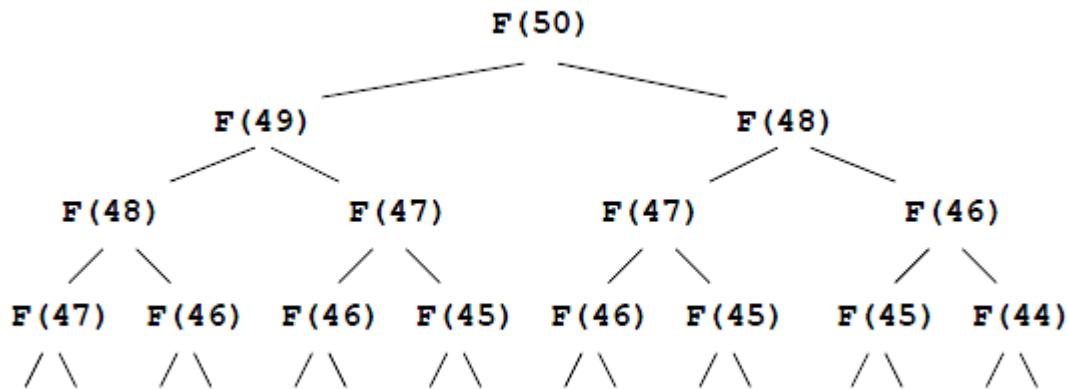
# اعداد فیبوناچی

```
def F(n):
    if (n<=1):
        return 1
    else:
        return F(n-1)+F(n-2)
```

$$F(a) = \begin{cases} 1 & a \leq 1 \\ F(a - 1) + F(a - 2) & a > 1 \end{cases}$$

آیا راه حل بازگشتی برای محاسبه  $F(50)$  کارا است؟

۱:  $F(50)$  بار فراخوانی می شود  
۱:  $F(49)$  بار فراخوانی می شود  
۲:  $F(48)$  بار فراخوانی می شود  
۳:  $F(47)$  بار فراخوانی می شود  
۵:  $F(46)$  بار فراخوانی می شود  
۸:  $F(45)$  بار فراخوانی می شود  
...  
۱۲۵۷۶۲۶۹۰۲۵:  $F(1)$  بار فراخوانی می شود



# اعداد فیبوناچی

آیا روش سریع‌تری برای محاسبه  $F(50)$  وجود دارد؟  
بله، کد زیر این کار را تنها با ۴۹ عمل جمع انجام می‌دهد.

```
def F(n):
    F=[1,1]
    for i in range(2,n):
        F.append(F[i-1]+F[i-2])
    return F
```

تحلیل و طراحی الگوریتم: تکنیک بالا برای محاسبه‌ی اعداد فیبوناچی، تکنیک برنامه‌ریزی پویا نام دارد.

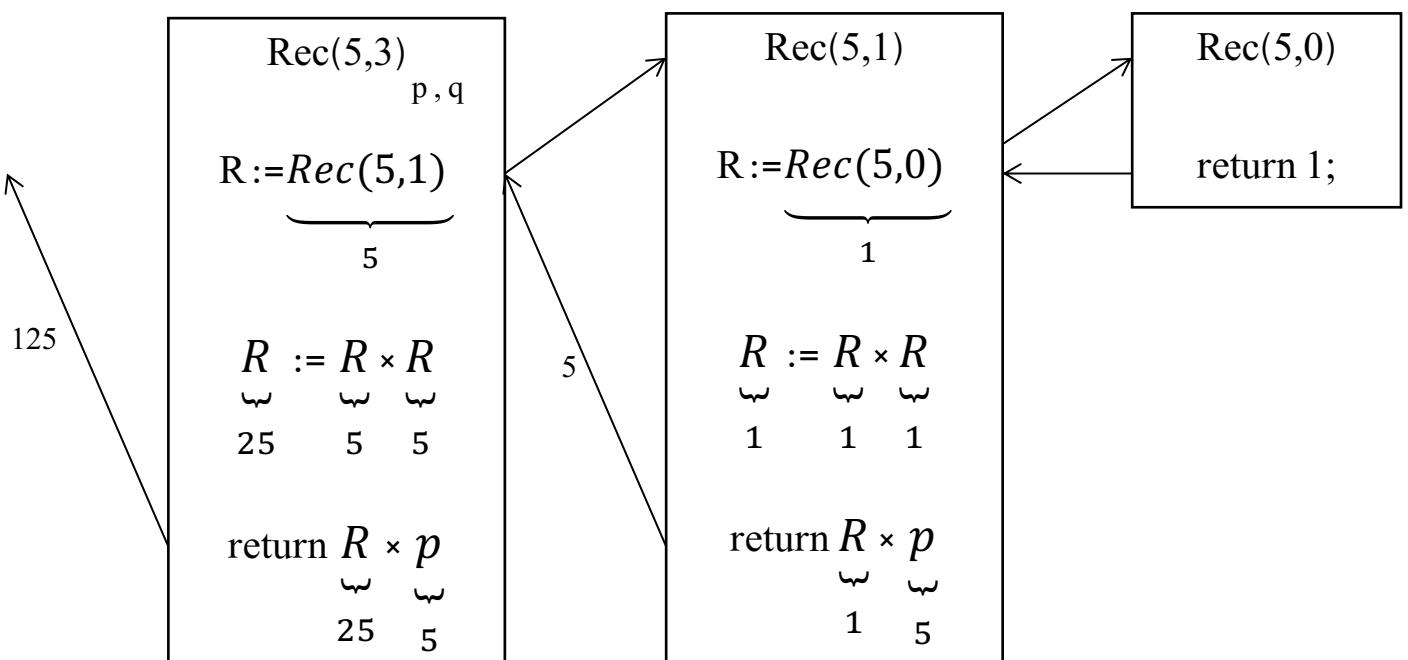
# مثال

```

def Rec(p,q):
    if(q<=0):
        return 1
    R=Rec(p,q/2)
    R=R*R
    if(q%2==0):
        return R
    else:
        return R*p

```

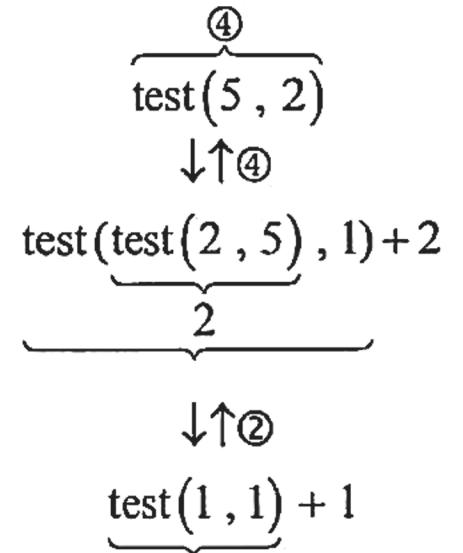
در روال بازگشتی زیر  $Rec(5,3)$  را محاسبه کنید؟



# مثال

```
def test(x,y):  
    if (x<=y or y==0):  
        t=x  
    elif (y==1):  
        t=test(x-1,y)+1  
    else:  
        t=test(test(y,x),y-1)+2  
    return t
```

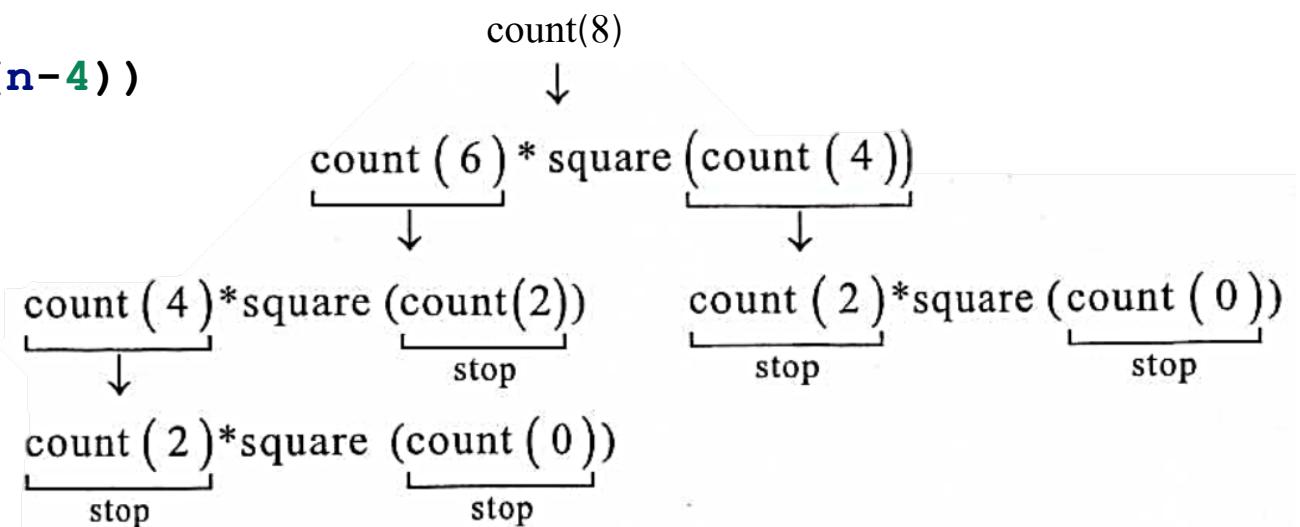
در روال بازگشتی زیر  $\text{test}(5,2)$  را محاسبه کنید؟



# مثال

```
def count(n):
    if(n<=0):
        return 1
    if(n==1):
        return 2
    if(n==2):
        return 3
    return count(n-2)*sqr(count(n-4))
```

در فراخوانی تابع زیر برای  $n=8$  به چند عمل ضرب نیاز است؟  
فرض کنید که هر فراخوانی  $\text{sqr}$  یک عمل ضرب نیاز دارد.



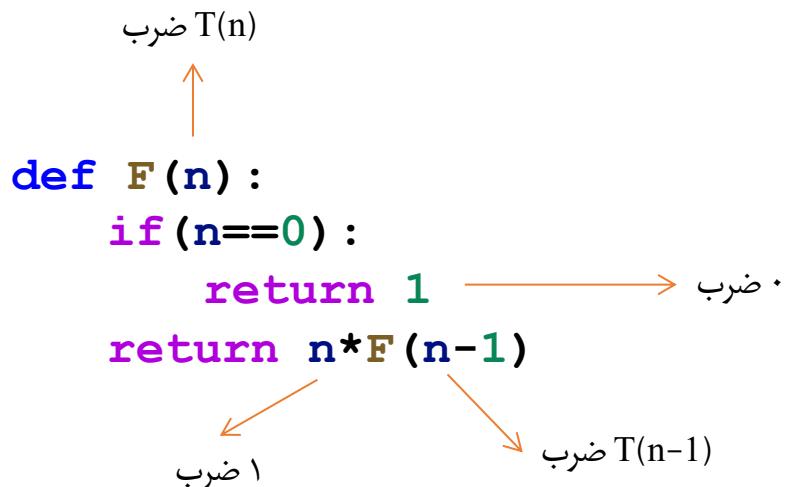
# تحلیل توابع بازگشتی

مراحل تحلیل یک الگوریتم بازگشتی

۱. تعیین عمل اصلی

۲. محاسبه‌ی تعداد دفعات اجرای عمل اصلی به صورت یک تابع بازگشتی از اندازه‌ی ورودی

۳. حل رابطه‌ی بازگشتی به دست آمده



# حل روابط بازگشتی

$$T(n) = \begin{cases} 0 & n = 0 \\ T(n - 1) + 1 & n > 0 \end{cases}$$

حدس و استقرا.

حدس جواب با استفاده از چند جمله اولیه  
اثبات حدس به وسیله استقرا

حدس

$$\begin{aligned} T(1) &= T(0) + 1 = 0 + 1 = 1 \\ T(2) &= T(1) + 1 = 1 + 1 = 2 \\ T(3) &= T(2) + 1 = 2 + 1 = 3 \\ T(4) &= T(3) + 1 = 3 + 1 = 4 \\ &\dots \\ T(n) &= n \end{aligned}$$

استقرا

$$\begin{aligned} \text{پایه به ازای } n=0 \text{ برقرار است} \\ T(0) &= 0 \\ \text{گام استقرا} \\ T(n + 1) &= T(n) + 1 = n + 1 \end{aligned}$$

# حل روابط بازگشتی

$$T(n) = \begin{cases} 1 & n = 1 \\ T(n/2) + 1 & n > 1, n = 2^k \end{cases}$$

حدس و استقرا.

حدس جواب با استفاده از چند جمله اولیه  
اثبات حدس به وسیله استقرا

حدس

$$\begin{aligned} T(2) &= T(1) + 1 = 1 + 1 = 2 \\ T(4) &= T(2) + 1 = 2 + 1 = 3 \\ T(8) &= T(4) + 1 = 3 + 1 = 4 \\ T(16) &= T(8) + 1 = 4 + 1 = 5 \\ \dots \\ T(n) &= \lg n + 1 \end{aligned}$$

استقرا

$$\begin{aligned} \text{پایه به ازای } n=1 \text{ برقرار است} \\ T(1) &= 1 + \lg 1 = 1 + 0 = 1 \\ \text{گام استقرا} \\ T(2n) &= T(n) + 1 \\ &= (\lg n + 1) + 1 \\ &= (\lg n + \lg 2) + 1 \\ &= \lg 2n + 1 \end{aligned}$$

# حل روابط بازگشتی

$$T(n) = \begin{cases} 1 & n = 1 \\ 2T(n - 1) + 1 & n > 1 \end{cases}$$

حدس و استقرا.

حدس جواب با استفاده از چند جمله اولیه  
اثبات حدس به وسیله استقرا

حدس

$$\begin{aligned} T(2) &= 2T(1) + 1 = 2 + 1 = 3 \\ T(2) &= 2T(2) + 1 = 6 + 1 = 7 \\ T(3) &= 2T(2) + 1 = 14 + 1 = 15 \\ T(4) &= 2T(3) + 1 = 30 + 1 = 31 \\ \dots \\ T(n) &= 2^n - 1 \end{aligned}$$

استقرا

$$\begin{aligned} \text{پایه به ازای } n=0 \text{ برقرار است} \\ T(0) &= 0 \\ T(n+1) &= 2T(n) + 1 \\ &= 2(2^n - 1) + 1 \\ &= 2^{n+1} - 2 + 1 \\ &= 2^{n+1} - 1 \end{aligned}$$

گام استقرا

# حل روابط بازگشتی

$$T(n) = \begin{cases} 0 & n = 1 \\ 2T(n/2) + n + 1 & n > 1, n = 2^k \end{cases}$$

حدس و استقرا.  
حدس جواب با استفاده از چند جمله اولیه  
اثبات حدس به وسیله استقرا

حدس

$$T(2) = 2T(1) + 1 = 0 + 1 = 1$$

$$T(4) = 2T(2) + 3 = 2 + 3 = 5$$

$$T(8) = 2T(4) + 7 = 10 + 7 = 17$$

$$T(16) = 2T(8) + 15 = 34 + 15 = 49$$

...

$$T(n) = ?$$

حدس راه حل همیشه هم ممکن نیست!

# حل روابط بازگشتی

$$T(n) = \begin{cases} 0 & n = 0 \\ T(n - 1) + 1 & n > 0 \end{cases}$$

جایگذاری مکرر.

$$\begin{aligned} T(n) &= T(n - 1) + 1 \\ &= T(n - 2) + 2 \\ &= T(n - 3) + 3 \\ &= T(n - 4) + 4 \\ &\quad \dots \\ &= T(n - i) + i \end{aligned}$$

$$\begin{aligned} T(n - i) &= T(0) && \Rightarrow n - i = 0 \\ &&& \Rightarrow i = n \\ T(n) &= T(n - n) + n \\ &= T(0) + n \\ &= 0 + n \\ &= n \end{aligned}$$

# حل روابط بازگشتی

$$T(n) = \begin{cases} 1 & n = 1 \\ T(n/2) + 1 & n > 1, n = 2^k \end{cases}$$

جایگذاری مکرر.

$$\begin{aligned} T(n) &= T(n/2^1) + 1 \\ &= T(n/2^2) + 2 \\ &= T(n/2^3) + 3 \\ &= T(n/2^4) + 4 \\ &\dots \\ &= T(n/2^i) + i \end{aligned}$$

$$\begin{aligned} T(n/2^i) &= T(1) && \Rightarrow n/2^i = 1 \\ &&& \Rightarrow i = \lg n \\ T(n) &= T(1) + \lg n \\ &= 1 + \lg n \end{aligned}$$

# حل روابط بازگشتی

$$T(n) = \begin{cases} 1 & n = 1 \\ 2T(n - 1) + 1 & n > 1 \end{cases}$$

جایگذاری مکرر.

$$\begin{aligned} T(n) &= 2^1 T(n-1) + 2^1 - 1 \\ &= 2^2 T(n-2) + 2^2 - 1 \\ &= 2^3 T(n-3) + 2^3 - 1 \\ &= 2^4 T(n-4) + 2^4 - 1 \\ &\quad \dots \\ &= 2^i T(n-i) + 2^i - 1 \end{aligned}$$

$$\begin{aligned} T(n - i) &= T(1) && \Rightarrow n - i = 1 \\ & && \Rightarrow i = n - 1 \\ T(n) &= 2^{n-1} T(1) + 2^{n-1} - 1 \\ &= 2 \cdot 2^{n-1} - 1 \\ &= 2^n - 1 \end{aligned}$$

# حل روابط بازگشتی

$$T(n) = \begin{cases} 0 & n = 1 \\ 2T(n/2) + n - 1 & n > 1, n = 2^k \end{cases}$$

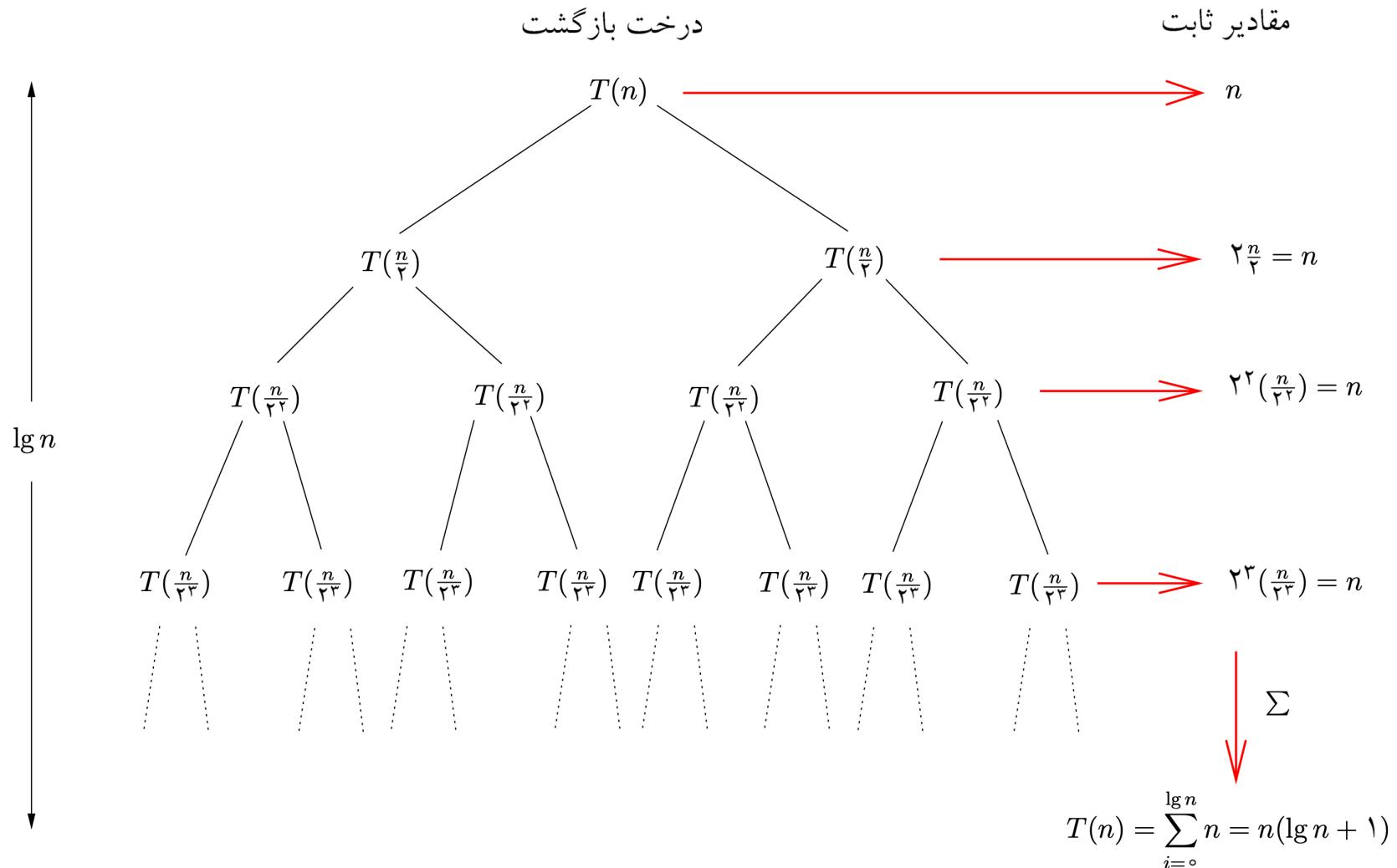
جایگذاری مکرر.

$$\begin{aligned} T(n) &= 2^1 T(n/2^1) + 1 \cdot n \\ &= 2^2 T(n/2^2) + 2 \cdot n \\ &= 2^3 T(n/2^3) + 3 \cdot n \\ &= 2^4 T(n/2^4) + 4 \cdot n \\ &\quad \dots \\ &= 2^i T(n/2^i) + i \cdot n \end{aligned}$$

$$\begin{aligned} T(n/2^i) &= T(1) && \Rightarrow n/2^i = 1 \\ &&& \Rightarrow i = \lg n \\ T(n) &= 2^{\lg n} T(1) + n \lg n \\ &= n \cdot 0 + n \lg n \\ &= n \lg n \end{aligned}$$

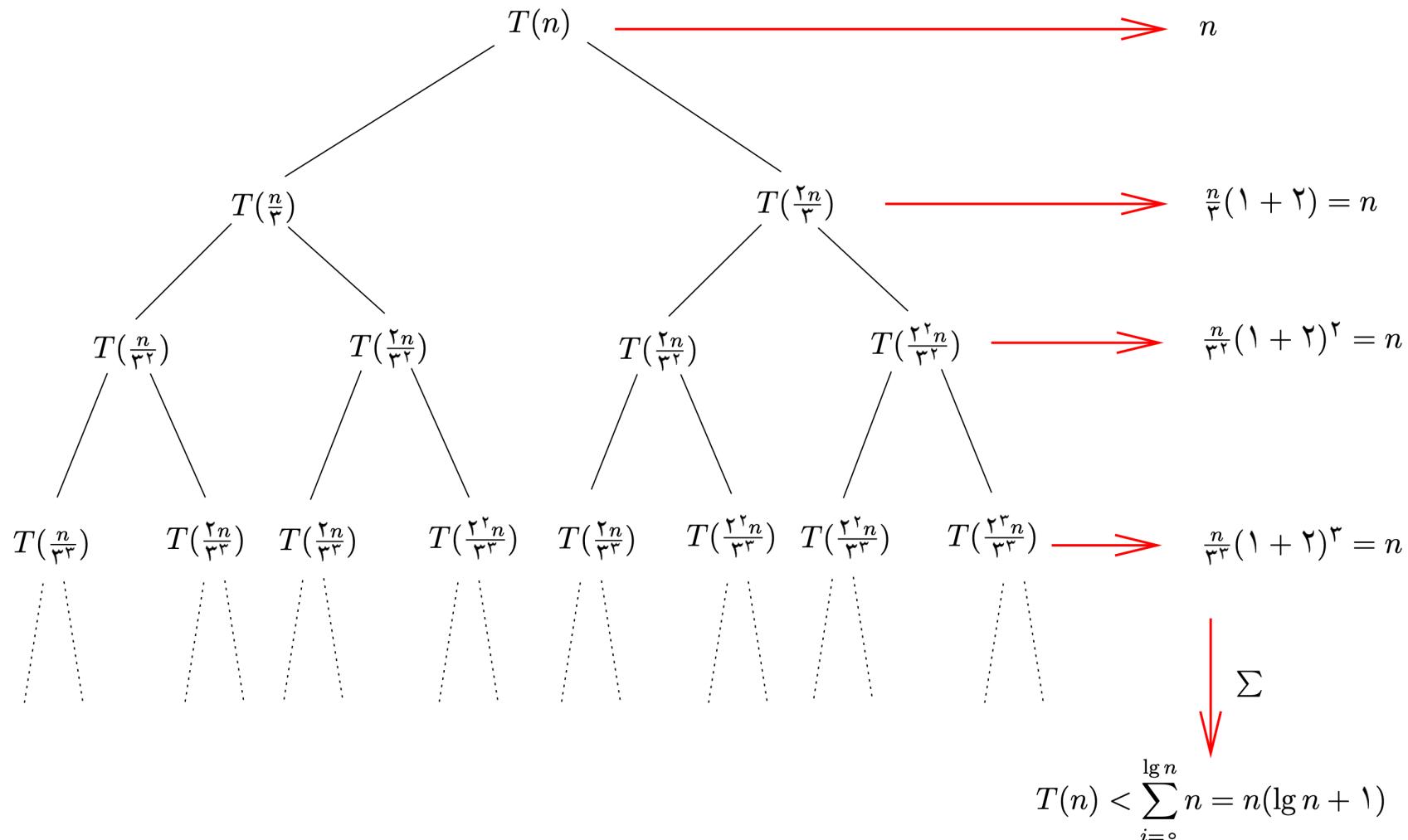
# درخت بازگشت

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



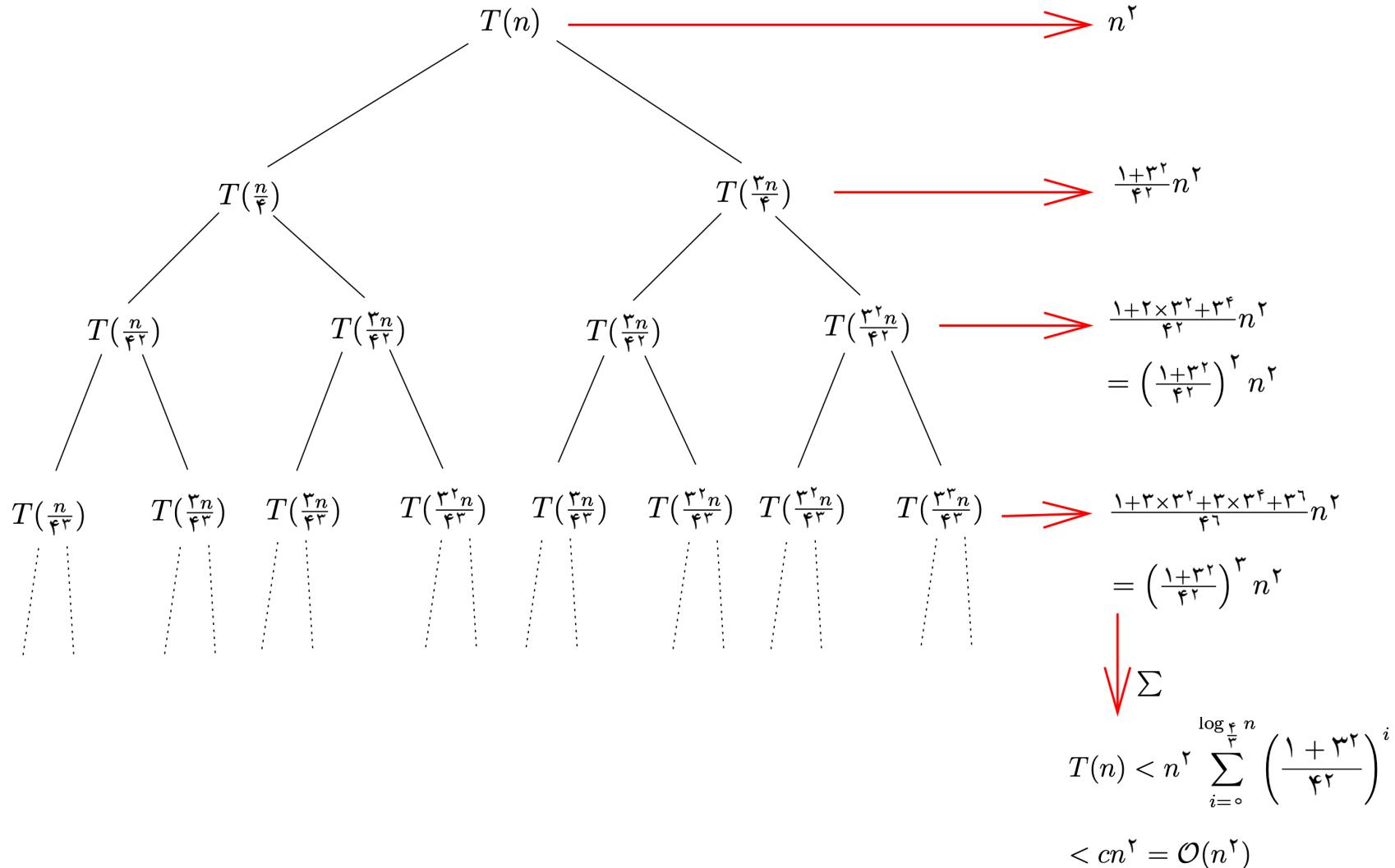
# درخت بازگشت

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



# درخت بازگشت

$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{3n}{4}\right) + n^2$$



# قضیه اصلی

مرتبه زمانی رابطه بازگشتی  $T(n)$  با مقایسه  $n^c$  و  $n^{\log_b^a}$  به یکی از سه حالت زیر محاسبه می‌شود:

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^c) \quad (a > 0, b > 1) \implies T(n) = \begin{cases} \Theta(n^{\log_b^a}) & \log_b^a > c \\ \Theta(n^c \lg n) & \log_b^a = c \\ \Theta(n^c) & \log_b^a < c \end{cases}$$

## قضیه اصلی

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

اگر  $n^{\log_b^a}$  بزرگتر باشد، آنگاه داریم:

$$(I) : f(n) = O(n^{\log_b a - \epsilon}) \xrightarrow{\epsilon > 0} T(n) = \Theta(n^{\log_b a})$$

$$T(n) = 4T\left(\frac{n}{2}\right) + \Theta(n^1) \implies T(n) = \Theta(n^2)$$

$$(a = 4, b = 2, c = 1) \Rightarrow \log_b a > c$$

## قضیه اصلی

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

اگر  $f(n)$  و  $n^{\log_b^a}$  صرف نظر از یک ضریب  $\log^k n$  مساوی باشند، آنگاه داریم:

$$(II) : f(n) = \Theta(n^{\log_a b} (\log n)^k) \xrightarrow{k \geq 0} T(n) = \Theta(n^{\log_b a} (\log n)^{k+1})$$

$$T(n) = 4T\left(\frac{n}{2}\right) + \Theta(n^2) \implies T(n) = \Theta(n^2 \lg n) \quad \text{مثال.}$$

$$(a = 4, b = 2, c = 2) \Rightarrow \log_b a = c$$

## قضیه اصلی

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

اگر  $f(n)$  بزرگتر باشد، آنگاه داریم:

$$(III) : f(n) = \Omega(n^{\log_b a + \epsilon}) \xrightarrow{\epsilon > 0} T(n) = \Theta(f(n))$$

$$T(n) = 4T\left(\frac{n}{2}\right) + \Theta(n^3) \implies T(n) = \Theta(n^3)$$

$$(a = 4, b = 2, c = 3) \Rightarrow \log_b a < c$$

# قضیه اصلی

**دقت کنید.** هنگام مقایسه  $f(n)$  و  $n^{\log_b^a}$ ، هدف از ذکر ع آن است که نشان دهیم تابع  $f(n)$  باید به صورت چندجمله ای با  $n^{\log_b^a}$  تفاوت داشته باشد، این یعنی مرتبه بزرگی یا کوچکی باید از  $n^\epsilon$  باشد، نه از مرتبه های  $\log n$  یا  $n \cos$  یا ...؛ زیرا در غیر این صورت قضیه اصلی در آن صدق نمی کند و استفاده از آن صحیح نمی باشد.

# مثال

$$T(n) = 9T\left(\frac{n}{3}\right) + n$$

$$\begin{cases} f(n) = n \\ n^{\log_b a} = n^{\log_3 9} = n^2 \simeq n^{1+\epsilon} \end{cases} \xrightarrow{f(n)=O(n^{\log_b a - \epsilon})} T(n) = \Theta(n^{\log_b a}) = \Theta(n^2)$$

# مثال

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

$$\begin{cases} f(n) = n \lg n \\ n^{\log_b a} = n^{\log_2 3} \simeq n^{1+\epsilon} \end{cases}$$

$$\xrightarrow{f(n)=O(n^{\log_b a - \epsilon})} T(n) = \Theta(n^{\log_b a}) = \Theta(n^{\log_2 3})$$

# مثال

$$T(n) = 7T\left(\frac{n}{2}\right) + n^2$$

$$\begin{cases} f(n) = n^2 \\ n^{\log_b a} = n^{\log_2 7} \simeq n^{2+\epsilon} \end{cases}$$

$$\xrightarrow{f(n)=O(n^{\log_b a - \epsilon})} T(n) = \Theta(n^{\log_b a}) = \Theta(n^{\log_2 7})$$

# مثال

$$T(n) = 100T\left(\frac{n}{99}\right) + \log n!$$

$$\begin{cases} f(n) = \log n! = \Theta(n \log n) \\ n^{\log_b a} = n^{\log_{99} 100} \simeq n^{1+\epsilon} \end{cases} \xrightarrow{f(n)=O(n^{\log_b a-\epsilon})} T(n) = \Theta(n^{\log_b a}) = \Theta(n^{\log_{99} 100})$$

# مثال

$$T(n) = 16T\left(\frac{n}{4}\right) + n^2$$

$$\begin{cases} f(n) = n^2 \\ n^{\log_b a} = n^{\log_4 16} = n^2 \end{cases}$$

$$\xrightarrow[k=0]{f(n)=\Theta(n^{\log_b a} (\log n)^k)} T(n) = \Theta(n^{\log_b a} (\log n)^{k+1}) = \Theta(n^2 \log n)$$

# مثال

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

ابتدا با استفاده از یک تغییر متغیر فرم تابع را به صورت یک قابل استفاده توسط قضیه اصلی می‌آوریم:

$$T(n) = T(\sqrt{n}) + 1 \xrightarrow{n=2^m} T(2^m) = T(2^{\frac{m}{2}}) + 1 \xrightarrow{S(m)=T(2^m)} S(m) = S\left(\frac{m}{2}\right) + 1$$

حال می‌توانیم برای رابطه  $S(m)$  از قضیه اصلی استفاده کنیم:

$$\begin{cases} f(m) = 1 \\ m^{\log_b a} = m^{\log_2 1} = m^0 = 1 \end{cases} \xrightarrow[k=0]{f(m)=\Theta(m^{\log_b a} (\log m)^k)} S(m) = \Theta(m^{\log_b a} (\log m)^{k+1}) = \Theta(\log m)$$

در پایان از آنجا که  $n=2^m$  در نظر گرفته بودیم، بنابراین  $\log n = m$  و خواهیم داشت:

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

# مثال

$$T(n) = 4T\left(\frac{\sqrt{n}}{3}\right) + \log^2 n$$

ابتدا با استفاده از یک تغییر متغیر فرم تابع را به صورت یک قابل استفاده توسط قضیه اصلی می‌آوریم:

$$T(n) = 4T\left(\frac{\sqrt{n}}{3}\right) + \log^2 n \xrightarrow{n=2^m} T(2^m) = 4T(2^{\frac{m}{2}}) + m^2 \xrightarrow{S(m)=T(2^m)} S(m) = 4S\left(\frac{m}{2}\right) + m^2$$

حال می‌توانیم برای رابطه  $S(m)$  از قضیه اصلی استفاده کنیم:

$$\begin{cases} f(m) = m^2 \\ m^{\log_b a} = m^{\log_2 4} = m^2 \end{cases} \xrightarrow[k=0]{f(m)=\Theta(m^{\log_b a} (\log m)^k)} S(m) = \Theta(m^{\log_b a} (\log m)^{k+1}) = \Theta(m^2 \log m)$$

در پایان از آنجا که  $n=2^m$  در نظر گرفته بودیم، بنابراین  $\log n = m$  و خواهیم داشت:

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

# مثال

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

$$\begin{cases} f(n) = n^2\sqrt{n} = n^{\frac{5}{2}} = n^{2.5} \\ n^{\log_b a} = n^{\log_2 4} = n^2 \end{cases} \xrightarrow[0 < \epsilon \leq 0.5]{f(n)=\Omega(n^{\log_b a+\epsilon})} T(n) = \Theta(f(n)) = \Theta(n^2\sqrt{n})$$

# مثال

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

$$\begin{cases} f(n) = n^3 \\ n^{\log_b a} = n^{\log_2 2} = n^1 \end{cases} \xrightarrow[0 < \epsilon \leq 2]{f(n)=\Omega(n^{\log_b a+\epsilon})} T(n) = \Theta(f(n)) = \Theta(n^3)$$

# مثال

$$T(n) = T\left(\frac{7n}{8}\right) + 8n + 1$$

$$\begin{cases} f(n) = 8n + 1 \\ n^{\log_b a} = n^{\log_{\frac{7}{8}} 1} = n^0 \end{cases} \xrightarrow[0 < \epsilon \leq 1]{f(n) = \Omega(n^{\log_b a + \epsilon})} T(n) = \Theta(f(n)) = \Theta(8n + 1) = \Theta(n)$$

# مثال

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

$$\begin{cases} f(n) = n^2 \\ n^{\log_b a} = n^{\log_2 3} \simeq n^{1.58} \end{cases} \xrightarrow[0 < \epsilon \leq 0.42]{f(n)=\Omega(n^{\log_b a + \epsilon})} T(n) = \Theta(f(n)) = \Theta(n^2)$$

# مثال

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

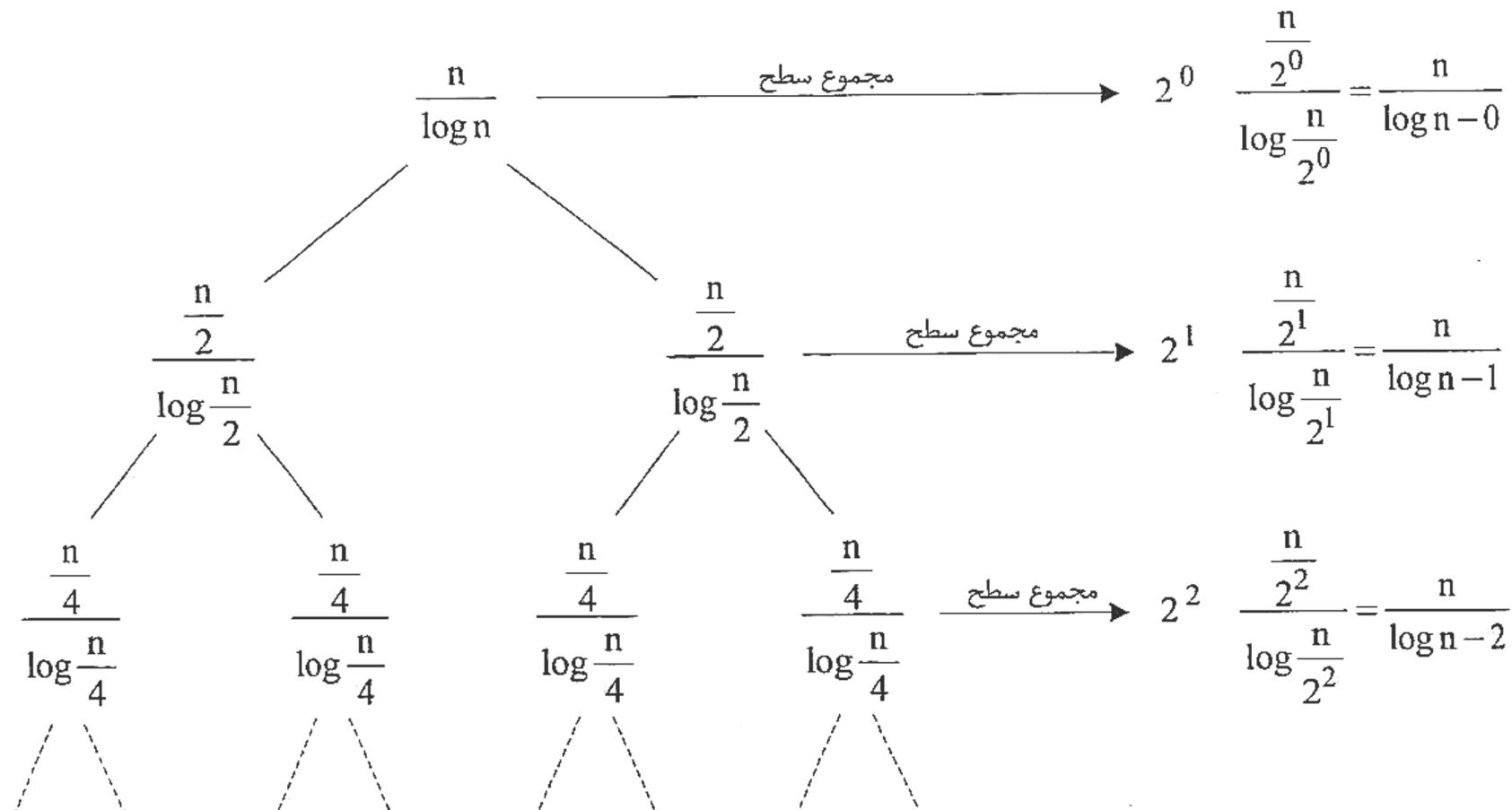
شاید در نگاه اول به نظر بیايد که این سوال از حالت (I) قضيه اصلی پیروی می کند:

$$\begin{cases} f(n) \frac{n}{\log n} \\ n^{\log_b a} = n^{\log_2 2} = n \end{cases}$$

چرا که  $n^{\log_b^a}$  بزرگتر از  $f(n)$  است، اما از آنجا که مرتبه  $n^{\log_b^a}$  از  $f(n)$  به صورت چند جمله ای بزرگتر نیست، پس نمی توان از قضيه اصلی استفاده کرد.

برای حل این مسئله می توانیم از درخت بازگشت استفاده کنیم.

# اداوه



# ادامه

$2^{\log n} = n$  باشند چرا که اگر به صورت  $\frac{2^{\frac{n}{\log n-1}}}{\log_2 \frac{n}{\log n}}$  هستند چرا که آنجایی که  $n$

سرابجام در سطح آخر گرهها به صورت  $\frac{2^{\frac{n}{\log n-1}}}{\log_2 \frac{n}{\log n-1}}$  است آن‌گاه به صورت  $\frac{1}{0} = \frac{1}{\log 1} = \infty$  می‌شوند که تعریف نشده است پس تعداد سطوح  $\log n$  از ۰ تا  $\log n - 1$  است و مجموع آن‌ها برابر است با:

$$\sum_{i=0}^{\log n - 1} \frac{n}{\log n - i} = n \left[ \underbrace{\frac{1}{\log n - 0}}_{\log n} + \underbrace{\frac{1}{\log n - 1}}_{2} + \dots + \underbrace{\frac{1}{\log n - (\log n - 2)}}_{1} + \underbrace{\frac{1}{\log n - (\log n - 1)}}_{1} \right]$$

$$= n \sum_{i=1}^{\log n} \frac{1}{i} = n \left[ \frac{1}{1} + \frac{1}{2} + \dots + \frac{1}{\log n} \right] = \boxed{\Theta(n \log \log n)}$$

# معادلات بازگشتی خطی

یک معادله بازگشتی به شکل  $c_0T(n) + c_1T(n-1) + \dots + c_kT(n-k) = 0$  که در آن  $k$  و  $c_i$  ها مقادیر ثابتی هستند، معادله بازگشتی خطی همگن با ضرایب ثابت نامیده می‌شود.

$$7T(n) - 3T(n-1) = 0$$

$$6T(n) - 5T(n-1) + 8T(n-2) = 0$$

$$8T(n) - 4T(n-3) = 0$$

تعریف: معادله شاخص برای معادله بازگشتی خطی همگن با ضرایب ثابت به صورت زیر تعریف می‌شود:

$$c_0r^k + c_1r^{k-1} + \dots + c_kr^0 = 0$$

برای مثال:

$$5T(n) - 7T(n-1) + 6T(n-2) = 0$$

$$5r^2 - 7r^1 + 6 = 0$$

در این مثال  $k=2$  است.

# معادلات بازگشتی خطی

قضیه: معادله بازگشتی خطی همگن با ضرایب ثابت به صورت زیر داده شده است:

$$c_0 T(n) + c_1 T(n-1) + \dots + c_k T(n-k) = 0$$

اگر معادله شاخص آن یعنی  $c_0 r^k + c_1 r^{k-1} + \dots + c_k r^0 = 0$  دارای  $k$  جواب مجزای  $r_1, r_2, \dots, r_k$  باشد، می‌توان نتیجه گرفت که تنها جواب معادله به صورت زیر است:

$$T(n) = c_1 r_1^n + c_2 r_2^n + \dots + c_k r_k^n$$

مقدار  $k$  و ثابت‌های  $c_i$  به وسیله وضعیت ابتدایی تعیین می‌شوند و برای تعیین این  $k$  ثابت منحصر به فرد به  $k$  وضعیت ابتدایی دنباله بازگشتی نیاز داریم.

# مثال

$$\begin{cases} T_n - 3T_{n-1} - 4T_{n-2} = 0 & , n > 1 \\ T_0 = 0 \\ T_1 = 1 \end{cases}$$

$$r^2 - 3r - 4 = 0 \quad (k = 2)$$

١. تعیین معادله شاخص

$$r^2 - 3r - 4 = (r - 4)(r - 1) = 0 \Rightarrow \begin{cases} r = 4 \\ r = -1 \end{cases}$$

٢. حل معادله شاخص

$$T_n = c_1 4^n + c_2 (-1)^n$$

٣. تعیین جواب عمومی معادله بازگشتی

$$\left. \begin{array}{l} T_0 = 0 = c_1 4^0 + c_2 (-1)^0 = c_1 + c_2 = 0 \\ T_1 = 1 = c_1 4^1 + c_2 (-1)^1 = 4c_1 - c_2 = 1 \end{array} \right\} \Rightarrow c_1 = \frac{1}{5}, c_2 = \frac{-1}{5}$$

٤. تعیین مقدار ثابت ها

$$T_n = \frac{1}{5} 4^n - \frac{1}{5} (-1)^n$$

٥. جواب نهایی

# مثال

$$\begin{cases} T_n - T_{n-1} - T_{n-2} = 0 & , n > 1 \\ T_0 = 0 \\ T_1 = 1 \end{cases}$$

$$r^2 - r - 1 = 0 \quad (k = 2)$$

١. تعیین معادله شاخص

$$r^2 - r - 1 = 0 \Rightarrow r = \frac{1 + \sqrt{5}}{2}, r = \frac{1 - \sqrt{5}}{2}$$

٢. حل معادله شاخص

$$T_n = c_1 \left( \frac{1 + \sqrt{5}}{2} \right)^n + c_2 \left( \frac{1 - \sqrt{5}}{2} \right)^n$$

٣. تعیین جواب عمومی معادله بازگشتی

$$T_0 = c_1 \left( \frac{1 + \sqrt{5}}{2} \right)^0 + c_2 \left( \frac{1 - \sqrt{5}}{2} \right)^0 = 0 \Rightarrow c_1 + c_2 = 0$$

٤. تعیین مقدار ثابت ها

$$T_1 = c_1 \left( \frac{1 + \sqrt{5}}{2} \right)^1 + c_2 \left( \frac{1 - \sqrt{5}}{2} \right)^1 = 1 \Rightarrow \left( \frac{1 + \sqrt{5}}{2} \right) c_1 + \left( \frac{1 - \sqrt{5}}{2} \right) c_2 = 1$$

## ادامه

$$c_1 = \frac{1}{\sqrt{5}}, c_2 = \frac{-1}{\sqrt{5}}$$

که با حل دستگاه دو معادله و دو مجهول فوق خواهیم داشت:  
۵. با جایگزینی ثابت ها در معادله خواهیم داشت:

$$T_n = \frac{\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n}{\sqrt{5}}$$

در قضیه ذکر شده باید تمامی  $k$  ریشه معادله شاخص از هم مجزا باشند، حال اگر معادله شاخص دارای ریشه های تکراری باشد، باید از قضیه دیگری استفاده کرد.

# معادلات بازگشتی خطی

قضیه: اگر  $r$  یک ریشه با توان  $m$  معادله شاخص برای یک معادله بازگشتی خطی همگن با ضرایب ثابت باشد، آنگاه

$$t_n = r^n, t_n = n r^n, t_n = n^2 r^n, \dots, t_n = n^{m-1} r^n$$

جواب های معادله بازگشتی هستند، بنابراین هر یک از این عبارات همانند قضیه قبل، یک عبارت در جواب عمومی معادله هستند.

# مثال

$$\begin{cases} T_n - 7T_{n-1} - 15T_{n-2} - 9T_{n-3} = 0 & , n > 2 \\ T_0 = 0 \\ T_1 = 1, T_2 = 1 \end{cases}$$

$$r^3 - 7r^2 - 15r - 9 = 0 \quad (k = 3)$$

١. تعیین معادله شاخص

$$r^3 - 7r^2 - 15r - 9 = (r - 1)(r - 3)^2 = 0 \Rightarrow r = 1, r = 3$$

٢. حل معادله شاخص

$$T_n = c_1 1^n + c_2 3^n + c_3 n 3^n$$

٣. تعیین جواب عمومی معادله بازگشتی

$$\left. \begin{array}{l} T_0 = 0 = c_1 + c_2 \\ T_1 = 1 = c_1 + 3c_2 + 3c_3 \\ T_2 = 2 = c_1 + 9c_2 + 18c_3 \end{array} \right\} \Rightarrow c_1 = -1, c_2 = 1, c_3 = \frac{-1}{3}$$

٤. تعیین مقدار ثابت ها

$$T_n = -1 + 3^n - n 3^{n-1}$$

٥. جواب نهایی

# مثال

```

for(i=1;i<=n;i++)
    for(j=1;j<=n;j++)
{
    x++;
    n--;
}

```

مرتبه زمانی شبہ کد زیر چیست؟

$$\sum_{i \in \mathbb{N}} \frac{n}{2^i} = n \sum_{i \in \mathbb{N}} \frac{1}{2^i} = n \left( \frac{\frac{1}{2}}{1 - \frac{1}{2}} \right) = n \in O(n)$$

i	j	n	تکرار	مجموع تکرار
1	1 2 ⋮ n/2	n → n-1 n → n-1 ⋮ n/2	1 1 ⋮ 1	n/2
2	1 2 ⋮ n/4	n/2 n/2 - 1 ⋮ n/4	1 1 ⋮ 1	n/4
3	1 2 ⋮ n/8	n/4 n/4 - 1 ⋮ n/8	1 1 ⋮ 1	n/8
⋮	⋮	⋮	⋮	⋮

# مثال

```

for(i=1;i<=n;i++)
{
    for(j=1;j<=n;j++)
        x++;
    n--;
}

```

$$\sum_{i \leq \frac{n}{2}} n - i = \frac{n^2}{2} - \sum_{i \leq \frac{n}{2}} i = \frac{n^2}{2} - \frac{\frac{n}{2}(\frac{n}{2} + 1)}{2} \in O(n^2)$$

مرتبه زمانی شبہ کد زیر چیست؟

i	j	n	تعداد تکرار حلقه با شمارنده j
1	1 ... n	n	n
2	1 ... n-1	n-1	n-1
3	1 ... n-2	n-2	n-2
:	:	:	:
n/2	1 ... n-(n/2)	n-(n/2)	n-(n/2)

# مثال

```

for(i=1;i<=n;i++)
    for(j=1;j<=n;j+=i)
        x++;
    
```

$$\sum_{i=1}^n \frac{n}{i} = n \sum_{i=1}^n \frac{1}{i} \in O(n \log n)$$

i	j	تکرار
1	1 2 ⋮ n	n
2	1 3 ⋮ n-1 یا n	n/2
3	1 4 ⋮ n-1 یا n-2 یا n	n/3
⋮	⋮	⋮
n-1	1 n	2
n	1	1