بسم الله الرحمن الرحیم

ساختمان‌های داده

جلسه ۶

مجتبی خلیلی
دانشکده برق و کامپیوتر
دانشگاه صنعتی اصفهان

مرتب‌سازی ادغامی

IUT-ECE

```
Merge ( Left, Right ) {
     nl=len(Left); nr=len(Right);  na=nl+nr;
     init (A,na);
     idl=0;  idr=0;

     for (i = 0; i < na; i++){
          if (Left[idl] > Right[idr] ) or (idl >= nl)
                A[i]= Right[idr];
                idr++;
          else
               A[i]= Left[idl];
               idl++;
     }
     return A;
   }
```

Does not work!

MERGE(A, p, q, r)

```
n1 = q - p + 1
n2 = r - q
let L[1 .. n1 + 1] and R[1 .. n2 + 1] be new arrays
for i = 1 to n1
    L[i] = A[p + i - 1]
for j = 1 to n2
    R[j] = A[q + j]
L[n1 + 1] = ∞
R[n2 + 1] = ∞
i = 1
j = 1
for k = p to r
    if L[i] ≤ R[j]
        A[k] = L[i]
        i = i + 1
    else A[k] = R[j]
        j = j + 1
```

```
MERGE(A, p, q, r)
n_L = q − p + 1             // length of A[p : q]
n_R = r − q                 // length of A[q + 1 : r]
let L[0 : n_L − 1] and R[0 : n_R − 1] be new arrays
for i = 0 to n_L − 1        // copy A[p : q] into L[0 : n_L − 1]
    L[i] = A[p + i]
for j = 0 to n_R − 1        // copy A[q + 1 : r] into R[0 : n_R − 1]
    R[j] = A[q + j + 1]
i = 0                       // i indexes the smallest remaining element in L
j = 0                       // j indexes the smallest remaining element in R
k = p                       // k indexes the location in A to fill
// As long as each of the arrays L and R contains an unmerged element,
//     copy the smallest unmerged element back into A[p : r].
while i < n_L and j < n_R
    if L[i] ≤ R[j]
        A[k] = L[i]
        i = i + 1
    else A[k] = R[j]
        j = j + 1
    k = k + 1
// Having gone through one of L and R entirely, copy the
//     remainder of the other to the end of A[p : r].
while i < n_L
    A[k] = L[i]
    i = i + 1
    k = k + 1
while j < n_R
    A[k] = R[j]
    j = j + 1
    k = k + 1
```

```
while i < n_L and j < n_R
    if L[i] ≤ R[j]
        A[k] = L[i]
        i = i + 1
    else A[k] = R[j]
        j = j + 1
    k = k + 1
// Having gone through one of L and R entirely, copy the
//     remainder of the other to the end of A[p : r].
while i < n_L
    A[k] = L[i]
    i = i + 1
    k = k + 1
while j < n_R
    A[k] = R[j]
    j = j + 1
    k = k + 1
```

حل رابطه بازگشتی

- چگونه به یک فرم بسته برای زمان اجرای الگوریتم مرتب‌سازی ادغامی برسیم؟

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \\ 2T(\frac{n}{2}) + cn & \text{otherwise} \end{cases}$$

حل رابطه بازگشتی

- به کمک بسط دادن:

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

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

$$= 4\left(2T\left(\frac{n}{8}\right) + \frac{cn}{4}\right) + 2cn = 8T\left(\frac{n}{8}\right) + 3cn$$

$$\ldots \qquad = 2^k T\left(\frac{n}{2^k}\right) + kcn$$

$$\text{Let } \frac{n}{2^k} = 1 \Rightarrow 2^k = n \Rightarrow k = \log_2 n$$

$$\Rightarrow T(n) = 2^k T\left(\frac{n}{2^k}\right) + kcn = n\,T(1) + cn\log_2 n$$

$$= \mathcal{O}(n \lg n)$$

مثال

- به کمک بسط دادن:

- $T(n) = T(n-3) + c$

$= T(n-2*3) + 2c = T(n-3*3) + 3c = T(n-4*3) + 4c$

$= \cdots = T(n-k*3) + kc$

$\Rightarrow n - 3k = 1 \Rightarrow k = \frac{n-1}{3}$

$\Rightarrow T(n) = T(n-3k) + kc = T(1) + c \ * \frac{n-1}{3} = \Theta(1) + \Theta(n) = \Theta(n)$

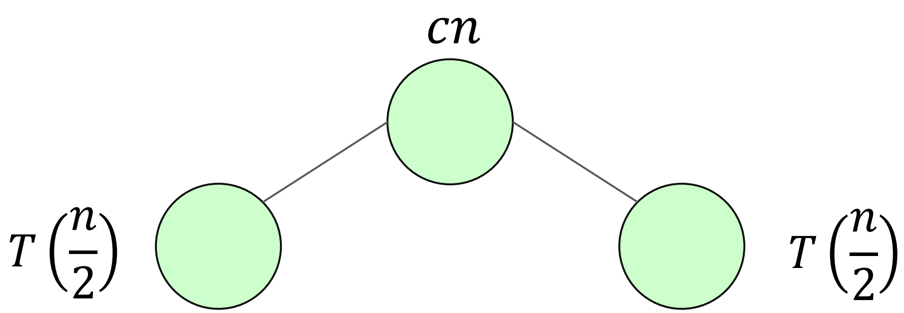
حل رابطه بازگشتی

- رابطه‌های بازگشتی را می‌توان به روش‌های زیر حل کرد:
    - حدس و استقراء (substitution method)
    - بسط دادن (Expanding)
    - درخت بازگشت (recursion-tree)
    - قضیه اصلی (Master Theorem)

حل رابطه بازگشتی

- درخت بازگشت:

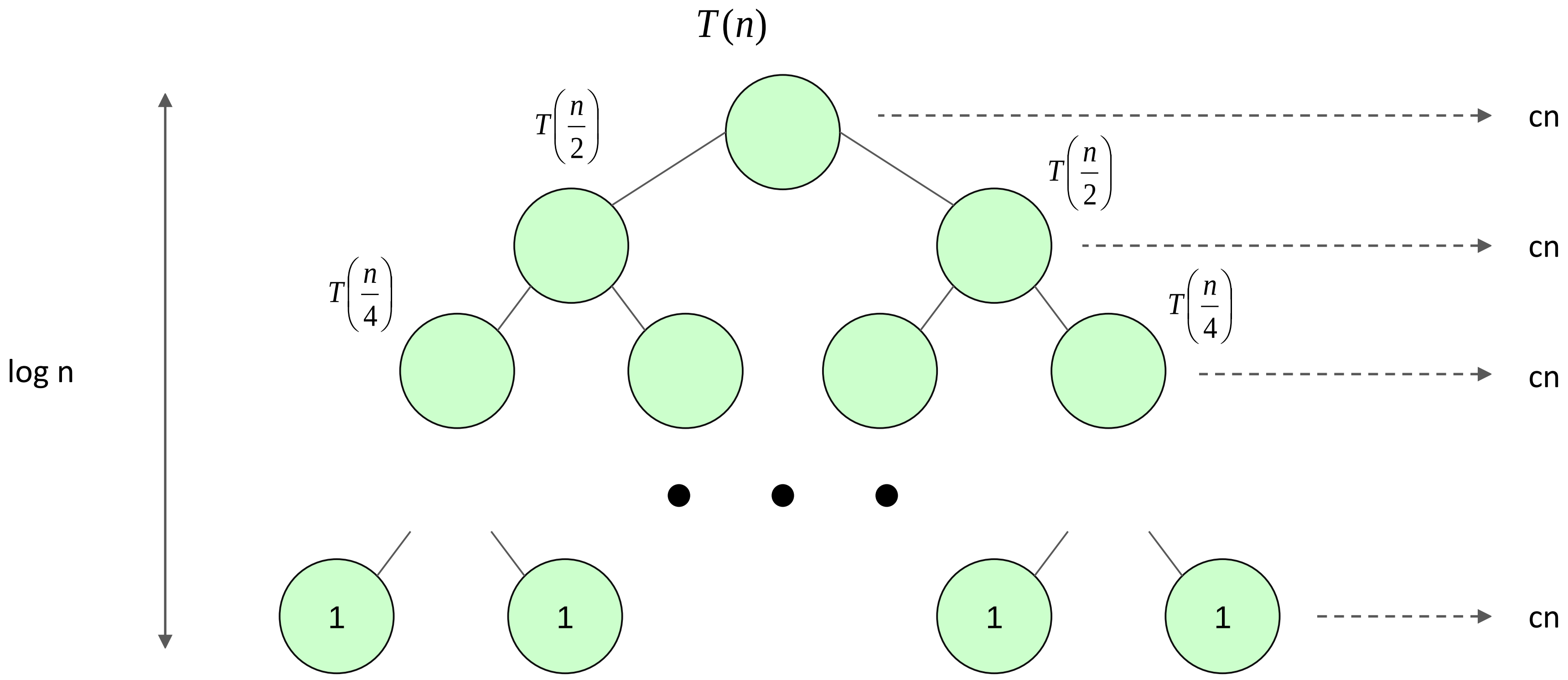
$$T(n) = 2T\left(\frac{n}{2}\right) + cn$$



حل رابطه بازگشتی

IUT-ECE

- درخت بازگشت:



$$T(n) = \Sigma_{i=0}^{\log n} cn = O(n \log n)$$

مثال

- یافتن یک کران بالا برای رابطه زیر

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

مثال

IUT-ECE

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

cn^2 $\qquad$ cn^2

$c\left(\frac{n}{4}\right)^2$ $\quad$ $c\left(\frac{n}{4}\right)^2$ $\quad$ $c\left(\frac{n}{4}\right)^2$ $\qquad$ $3/16cn^2$

$c\left(\frac{n}{16}\right)^2$ $c\left(\frac{n}{16}\right)^2$ $c\left(\frac{n}{16}\right)^2$ $\quad$ $c\left(\frac{n}{16}\right)^2$ $c\left(\frac{n}{16}\right)^2$ $c\left(\frac{n}{16}\right)^2$ $\quad$ $c\left(\frac{n}{16}\right)^2$ $c\left(\frac{n}{16}\right)^2$ $c\left(\frac{n}{16}\right)^2$ $\qquad$ $(3/16)^2cn^2$

هزینه در هر سطح: $\left(\frac{3}{16}\right)^d cn^2$

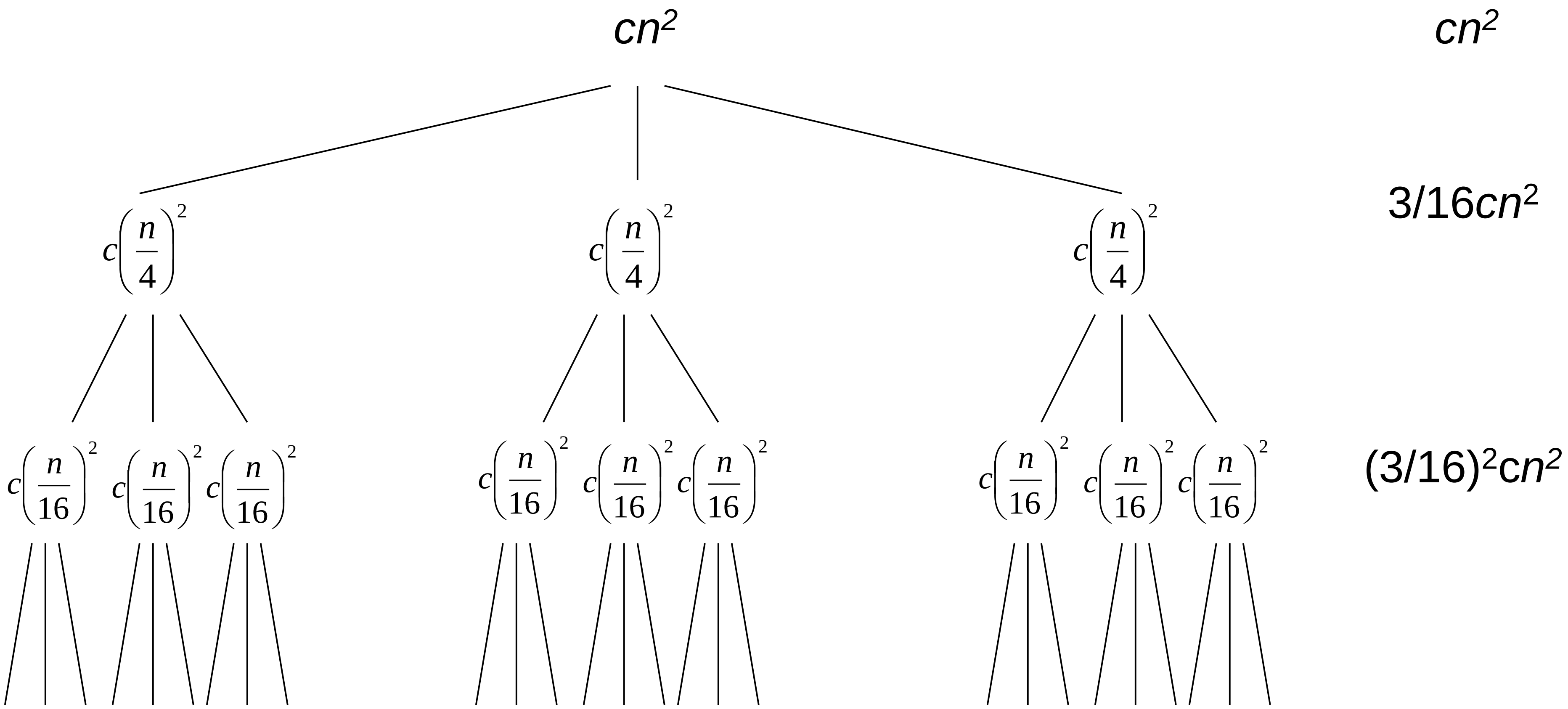
مثال

- در هر سطح n تقسیم بر ۴ می‌شود:

$$\frac{n}{4^d} = 1$$

$$d = \log_4 n$$

مثال



هزینه در آخرین سطح: $3^d = 3^{\log_4 n}$

مثال

- زمان کل:

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

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

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

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

$$= \frac{16}{13} cn^2 + \Theta(3^{\log_4 n})$$

مثال

- زمان کل:

$$T(n) = \frac{16}{13}cn^2 + \Theta(3^{\log_4 n})$$

$$3^{\log_4 n} = 4^{\log_4 3^{\log_4 n}}$$

$$= 4^{\log_4 n \log_4 3}$$

$$= 4^{\log_4 n^{\log_4 3}}$$

$$= n^{\log_4 3}$$

$$T(n) = \frac{16}{13}cn^2 + \Theta(n^{\log_4 3})$$

$$T(n) = O(n^2)$$