بسم الله الرحمن الرحیم

ساختمان‌های داده

جلسه ۹

مجتبی خلیلی
دانشکده برق و کامپیوتر
دانشگاه صنعتی اصفهان

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

- قضیه اصلی: فرض کنید $a \geq 1$ و $b > 1$ و p یک ثابت باشد.
همچنین فرض کنید $T(n) = a \cdot T\left(\frac{n}{b}\right) + O(n^p)$.
آنگاه:

$$T(n) = \begin{cases} O(n^p \log(n)) & \text{if } a = b^p \\ O(n^p) & \text{if } a < b^p \\ O\left(n^{\log_b(a)}\right) & \text{if } a > b^p \end{cases}$$

- مقدار $\frac{n}{b}$ می‌تواند $\lfloor n/b \rfloor$ یا $\lceil n/b \rceil$ نیز باشد.

حل رابطه بازگشتی

- قضیه اصلی برای مرتب‌سازی ادغامی:

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

a = 2
b = 2
p = 1

$a = b^p$

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

مثال

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

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

مثال

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

$$T(n) = O(n^{\log_2 3})$$

مثال

$$T(n) = T\left(\frac{2n}{3}\right) + 1$$

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

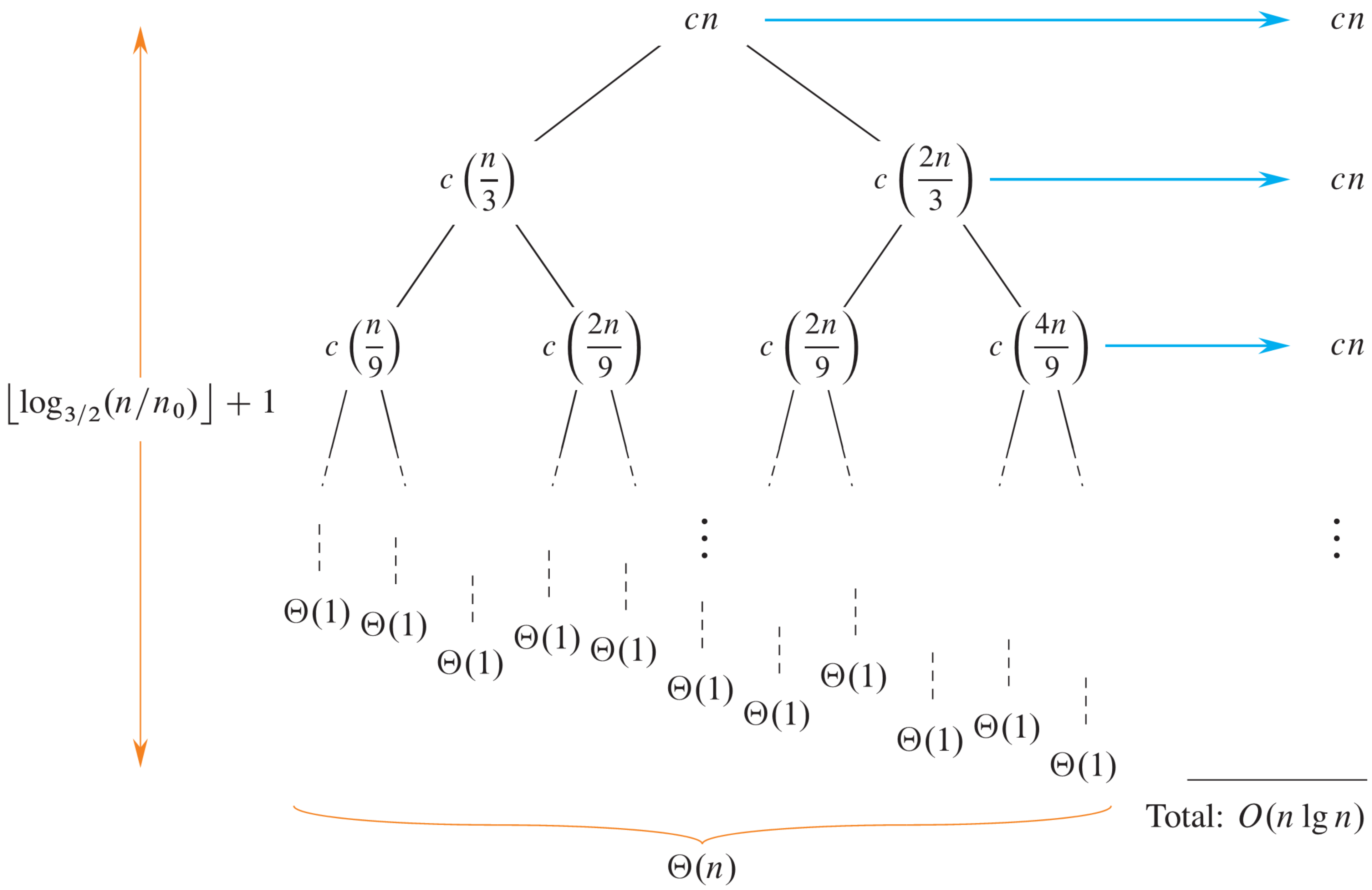
استفاده از قضیه اصلی

- این قضیه همیشه قابل استفاده نیست.

مثال

- کران بالا برای رابطه بازگشتی زیر

$$T(n) = T(n/3) + T(2n/3) + \Theta(n) .$$



cn
$c\left(\frac{n}{3}\right)$
$c\left(\frac{2n}{3}\right)$
$c\left(\frac{n}{9}\right)$
$c\left(\frac{2n}{9}\right)$
$c\left(\frac{2n}{9}\right)$
$c\left(\frac{4n}{9}\right)$
cn
cn
cn
$\lfloor \log_{3/2}(n/n_0) \rfloor + 1$
$\Theta(1)$
$\Theta(n)$
Total: $O(n \lg n)$

مسئله جستجو

Input: A sequence of n numbers $\langle a_1, a_2, \ldots, a_n \rangle$ stored in array $A[1:n]$ and a value x.

Output: An index i such that x equals $A[i]$ or the special value NIL if x does not appear in A.

جستجوی خطی

- یک الگوریتم برای حل مسئله:

```
int linSearch(int arr [], int x) {
   for (int i = 0; i < arr.length; i++) {
      if (arr[i] == x)
            return i;
   }
   return -1;
}
```

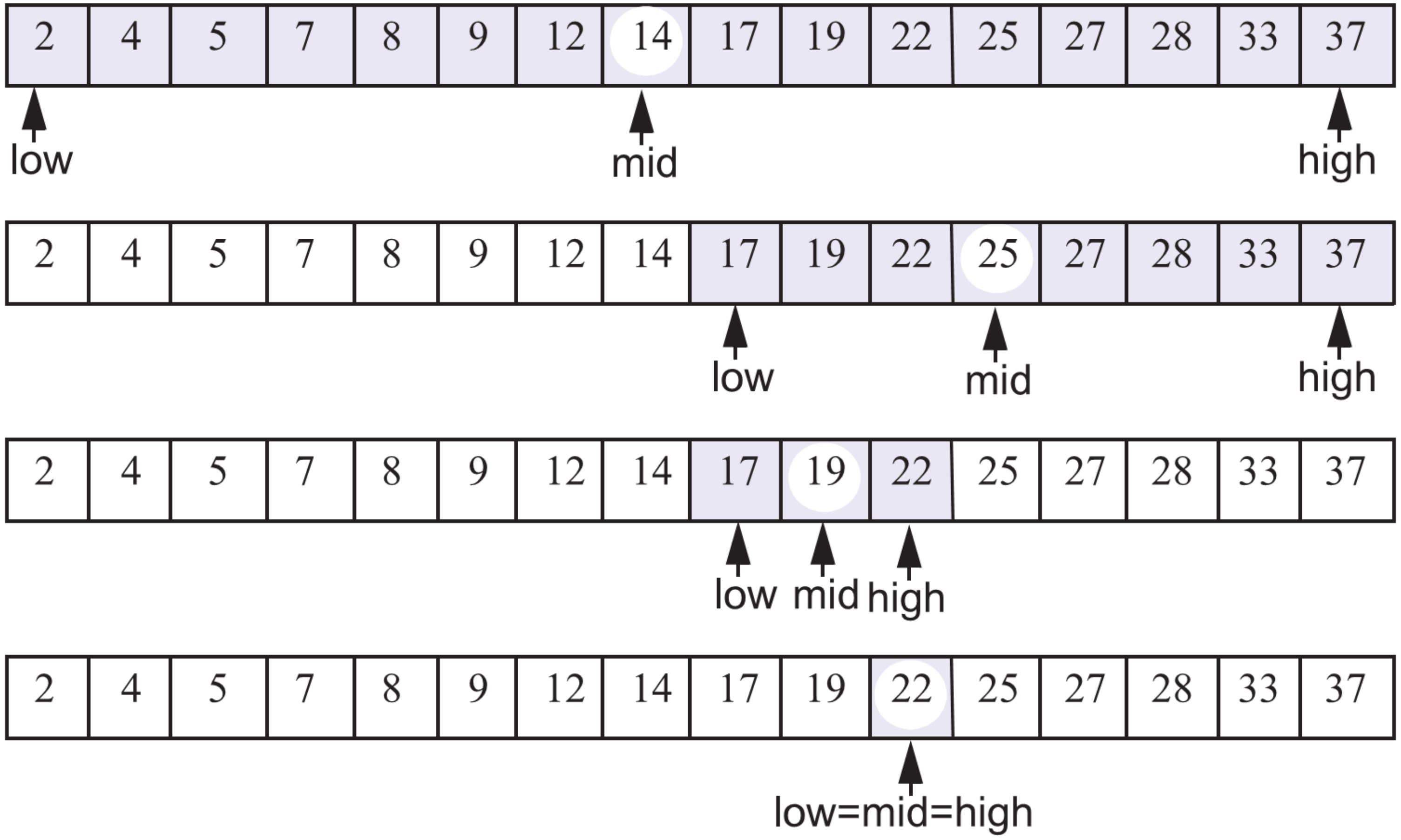
- پیچیدگی الگوریتم؟

مسئله جستجو

- الگوریتم دیگر برای حل مسئله؟
- انجام پیش پردازش

مسئله جستجو

- اگر آرایه مرتب شده باشد؟ مثلا پیدا کردن عدد ۲۲



جستجوی دودویی

IUT-ECE

```
int binarySearch(int arr[], int x, int low, int high) {
     if( high < low ) {
         return -1;
     } if(high == low) {
         if(arr[high] == x) {
             return high;
         }
         return -1;
     }
     int mid = (low+high) / 2;
     if(arr[mid] == x) {
         return mid;
     } else if(arr[mid] < x) {
         return binarySearch(arr, x, mid+1, high);
     } else {
         return binarySearch(arr, x, low, mid-1);
     }
 }
```

+6

بازگشتی

تحلیل رابطه بازگشتی

- اکنون به طور دقیقتر این رابطه را تحلیل می‌کنیم:

$$T(n) = \begin{cases} d & n = 1, \\ T(n/2) + c & n > 1. \end{cases}$$

$$\begin{aligned} T(n) &= T\left(\frac{n}{2}\right) + c = T\left(\frac{n}{4}\right) + 2c = \ldots \\ &= T\left(\frac{n}{2^i}\right) + i \cdot c \\ &= T\left(\frac{n}{n}\right) + \log_2 n \cdot c = d + c \cdot \log_2 n \in \Theta(\log n) \end{aligned}$$

مسئله جستجو

IUT-ECE

- فرض کنید طول آرایه برابر n است و قصد m درخواست برای جستجو داریم.
  - جستجوی خطی
    - پیش‌پردازش
      - ندارد، $O(1)$
    - جستجو
      - $O(n)$
  - زمان کل
    - $O(1) + m \times O(n) = O(mn)$
  - $m = n$
    - $O(n^2)$

مسئله جستجو

- فرض کنید طول آرایه برابر n است و قصد m درخواست برای جستجو داریم.
  - جستجوی دودویی
    - پیش‌پردازش
      - مرتب‌سازی، $O(n \log n)$
    - جستجو
      - $O(\log n)$
  - زمان کل

$$O(n \log n) + m \times O(\log n) = O\big((n+m) \log n\big)$$

  - $m = n$
  - $O(n \log n)$