课程回顾

- 求解递归式（计算时间复杂度）：递归树法、主方法
- 分治法的适用条件：规模缩小到一定程度易解决、最优子结构、子问题解可合并、分解出的子问题相互独立
- 分治法求解应用：二分搜索、最大子数组、矩阵乘法Strassen算法、大整数乘法、快速排序

分治法求解——快速排序

- （教材p95）快速排序尽管最坏情况时间复杂度是$\Theta(n^2)$，但平均情况时间复杂度为$\Theta(n\lg n)$
- 基于比较的排序算法时间下界为：

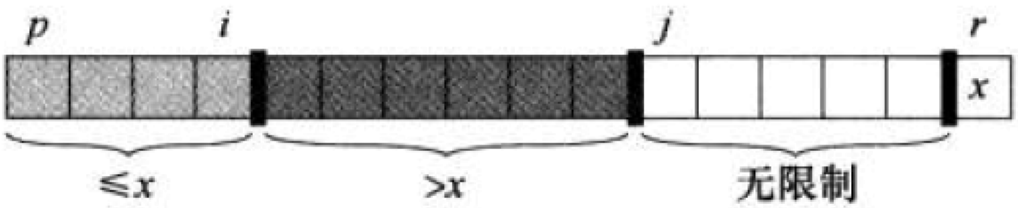
$$\begin{aligned}\lg(n!) &= \lg(\sqrt{2\pi n}(n/e)^n(1+\Theta(1/n))) \\ &= \lg(2\pi n)/2 + n\lg(n/e) + \lg(1+\Theta(1/n)) \\ &= \lg(2\pi)/2 + (\lg n)/2 + n\lg n - n\lg e + \lg(1+\Theta(1/n)) \\ &\approx n\lg n - 1.44n + \Theta(\lg n)\end{aligned}$$

- 快速排序平均情况：$1.39n\lg n+O(n)$，因系数较小称为“快排”

分治法求解——快速排序 (续)

■算法描述

```
QUICKSORT(A, p, r)
if p < r
    q ← PARTITION(A, p, r)
    QUICKSORT(A, p, q-1)
    QUICKSORT(A, q+1, r)
```




```
PARTITION(A, p, r)
x ← A[r]  //划分元/主元(pivot element)
i ← p - 1
for j ← p to r-1 do
    if A[j] ≤ x
        i ← i + 1,  exchange A[i] with A[j]
exchange A[i+1] with A[r]
return i+1
```

循环不变式：

1. $A[p..i] \le x$
2. $A[i+1..j-1] > x$
3. $A[r] = x$

该划分使得：

$A[p..q-1] \le A[q] < A[q+1..r]$

分治法求解——快速排序 (续)

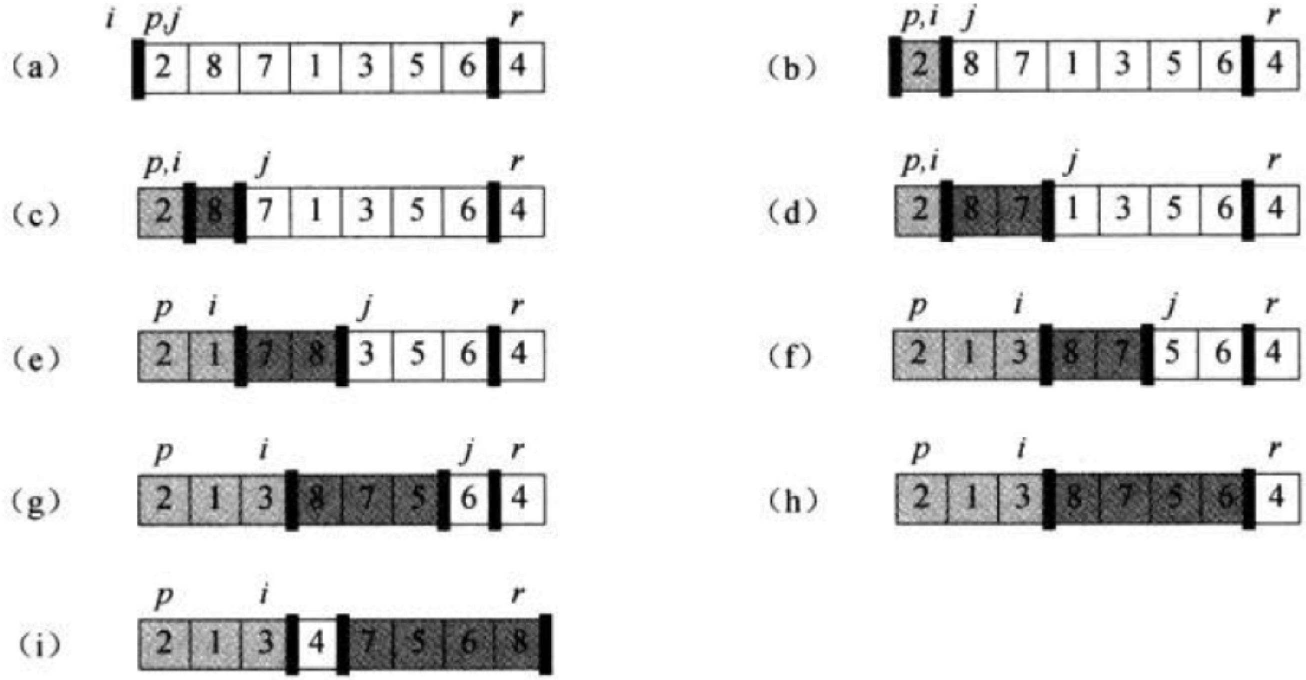


图 7-1　在一个样例数组上的 PARTITION 操作过程。数组项 $A[r]$是主元 x。浅阴影部分的数组元素都在划分的第一部分，其值都不大于 x。深阴影部分的元素都在划分的第二部分，其值都大于 x。无阴影的元素则是还未分入这两个部分中的任意一个。最后的白色元素就是主元 x。(a)初始的数组和变量设置。数组元素均未被放入前两个部分中的任何一个。(b)2 与它自身进行交换，并被放入了元素值较小的那个部分。(c)～(d)8 和 7 被添加到元素值较大的那个部分中。(e)1 和 8 进行交换，数值较小的部分规模增加。(f)数值 3 和 7 进行交换，数值较小的部分规模增加。(g)～(h)5 和 6 被包含进较大部分，循环结束。(i)在第 7～8 行中，主元被交换，这样主元就位于两个部分之间

分治法求解——快速排序 (续)

■性能分析：划分是否平衡？

- 最坏情况划分（已经有序排列）

$$\begin{aligned}T(n) &= T(n-1) + T(0) + \Theta(n) \\ &= T(n-1) + \Theta(n) \\ &= \sum_{k=1}^{n} \Theta(k) = \Theta(\sum_{k=1}^{n} k) = \Theta(n^2)\end{aligned}$$

- 最好情况划分（两子问题大小大致相等）

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

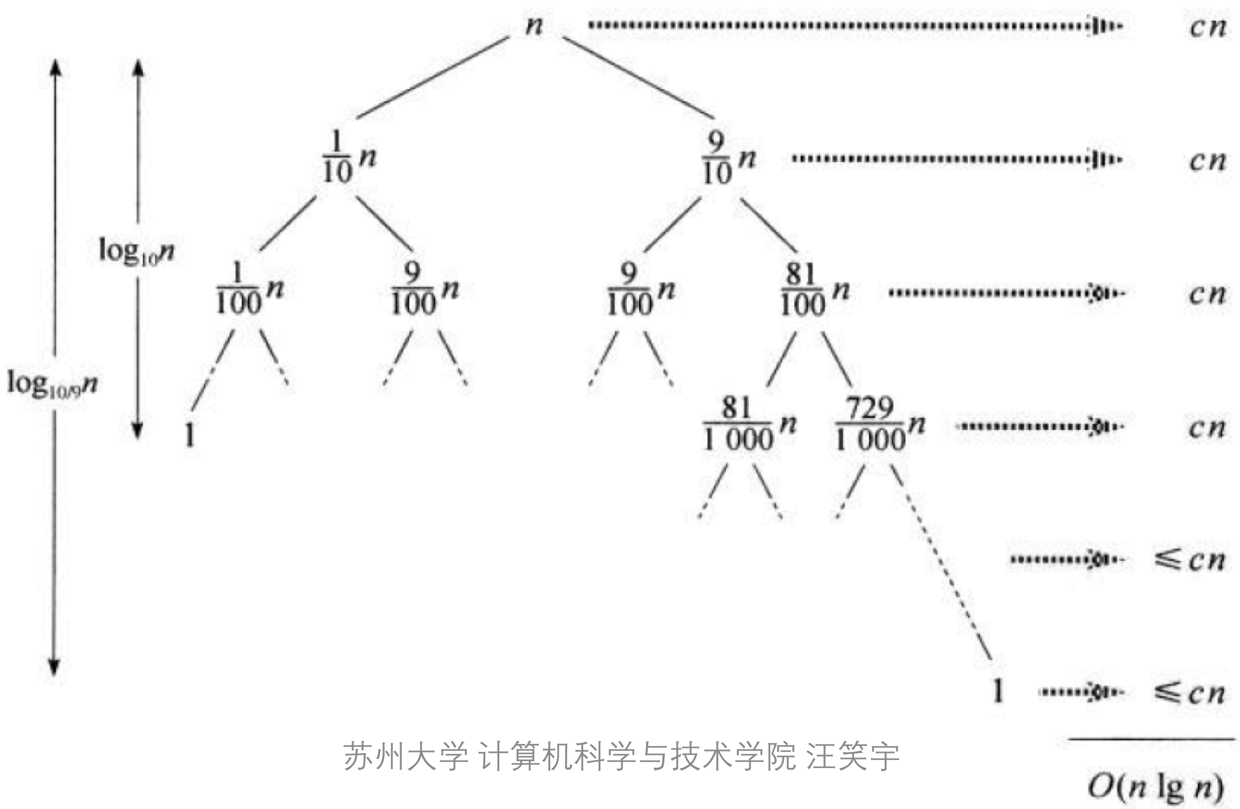
分治法求解——快速排序 (续)

■性能分析：划分是否平衡？

➢平均情况划分（平均情况时间接近于最好情况时间）

- 假设划分总是产生9:1划分

$$T(n) \leq T(9n/10) + T(n/10) + cn$$



分治法求解——快速排序 (续)

- 性能分析：划分是否平衡？
  - 平均情况划分（平均情况时间接近于最好情况时间）
    - 因为任何底大于1的对数与以2为底的对数之间只相差一个常数因子
    - 所以任何常数比例的划分，树高仍为$O(\lg n)$，从而快排时间为$O(n\lg n)$

分治法求解——快速排序 (续)

■随机化版本（教材p100）

- 快速排序的平均性能假定：输入的所有排列是等可能的
- 算法随机化是指：算法行为不仅由输入确定，而且与随机数发生器产生的值有关，强迫输入分布是随机的

RANDOMIZED_PARTITION(A, p, r)
1 $i \leftarrow$ RANDOM(p, r)
2 exchange $A[r]$ with $A[i]$
3 **return** PARTITION(A, p, r)

- 分析较困难
- 算法非常有效，排序过程中，某次随机选择最坏不会影响总体效果

分治法求解——期望为线性时间的选择算法

■求n个元素组成的序列中第i小的元素（假设所有元素互异）

```
RANDOMIZED_SELECT(A, p, r, i)
if p = r  return A[p]
q ← RANDOMIZED_PARTITION(A, p, r)
k ← q – p + 1
if i = k return A[q]
elseif i < k return RANDOMIZED_SELECT(A, p, q–1, i)
else return RANDOMIZED_SELECT(A, q+1, r, i–k)
```

```
RANDOMIZED_PARTITION(A, p, r)
i ← RANDOM(p, r)
exchange A[r] with A[i]
return PARTITION(A, p, r)
```

分治法求解——期望为线性时间的选择算法 (续)

- 平均情况：$T(n) = T(n/2) + \Theta(n)$
  - 由主方法第三种情况可得：$T(n) = \Theta(n)$
- 最坏情况：$T(n) = T(n-1) + \Theta(n)$
  - 与快速排序最坏情况相同，为$\Theta(n^2)$