Lec06 顺序统计、中值

Order Statistics

given n elements in array (unsorted), to find the k-th smallest element

naive algorithm: to sort, and then return the k-th element

$k=1$ minimum

$k=n$ maximum

$k=\lfloor \frac{n+1}{2} \rfloor$ or $\lceil \frac{n+1}{2} \rceil$ medians (typically)

randomized divide and conquer algorithm:

(pseu-code)
```
Rand-Select (A, p, q, i)  // to find the i-th smallest in A[p...q]   "找第i小的元素"
    if p = q then
        return A[p]
    r ← Rand-Partition(A, p, q)
    k ← r-p+1       // A[r] is the k-th smallest element in A[p...q]
    if i = k then return A[r]
    elif i < k then
        return Rand-Select (A, p, r-1, i)
    else (i > k) then
        return Rand-Select (A, r+1, q, i-k)
```

Example:

$A = 6, 10, 13, 5, 8, 3, 2, 11$, $i = 7$

$A =$

6	10	13	5	8	3	2	11

pivot

$A' =$

2	5	3	6	8	13	10	11

$r=4$, "意味着如果我们一开始对A进行排序，6一定会在index=4的位置(从1开始)"，即"6是第4小的元素"。

$k = r-p+1 = 4-1+1 = 4 < 7 = i$

$A' =$

2	5	3	6	8	13	10	11

"找这里第 $i-k=3$ 大的元素"

Intuition for Analysis:

(today assume distinct elements)

lucky case: (1/10 : 9/10 for example)

$$T(n) \le T(\tfrac{9}{10}n) + \Theta(n)$$
$$= \Theta(n)$$

unlucky case: (0 : n-1)

$$T(n) = T(n-1) + \Theta(n)$$
$$= \Theta(n^2) \quad \text{"arithmetic"}$$

Analysis of Expected Time:

- let $T(n)$ be the random variable for running time of Random-Select on an input of size n, assuming random numbers are chosen independently
- define indicator random variable X_k $(k = 0, 1, 2, \cdots, n-1)$.

$$X_k = \begin{cases} 0, & \text{otherwise} \\ 1, & \text{if the partition comes out that } k \text{ on the left-hand side } (k : n-k-1 \text{ split}) \end{cases}$$

$$T(n) \leqslant \begin{cases} T(\max\{0, n-1\}) + \Theta(n), & \text{if } 0 : n-1 \text{ split} \\ T(\max\{1, n-2\}) + \Theta(n), & \text{if } 1 : n-2 \text{ split} \\ \cdots \quad \cdots \quad \cdots \quad \cdots & \cdots \\ T(\max\{n-1, 0\}) + \Theta(n), & \text{if } n-1 : 0 \text{ split} \end{cases}$$

$$= \sum_{k=0}^{n-1} X_k \cdot \left[T(\max\{k, n-1-k\}) + \Theta(n) \right]$$

$$E(T(n)) = E\left(\sum_{k=0}^{n-1} X_k \cdot \left[T(\max\{k, n-1-k\}) + \Theta(n) \right] \right)$$

$$= \sum_{k=0}^{n-1} E\left(X_k \cdot \left[T(\max\{k, n-1-k\}) + \Theta(n) \right] \right)$$ "递归中每次产生的随机数，都独立于首次调用时产生的随机数"

$$= \sum_{k=0}^{n-1} E(X_k) \cdot E\left(T(\max\{k, n-1-k\}) + \Theta(n) \right)$$

$$= \sum_{k=0}^{n-1} \frac{1}{n} \cdot E\left(T(\max\{k, n-1-k\}) + \Theta(n) \right)$$

$$= \frac{1}{n} \sum_{k=0}^{n-1} E\left(T(\max\{k, n-1-k\}) \right) + \frac{1}{n} \sum_{k=0}^{n-1} \Theta(n)$$

$$\leqslant \frac{2}{n} \sum_{k=\lfloor n/2 \rfloor}^{n-1} E(T(k)) + \Theta(n)$$

claim: $E(T(n)) \leqslant c \cdot n$ for sufficiently large constant $c > 0$

proof: (substitution method)

assume true for "$< n$"

$$E(T(n)) \leqslant \frac{2}{n} \sum_{k=\lfloor n/2 \rfloor}^{n-1} \underbrace{E(T(k))}_{\text{“} \leqslant ck \text{, by hypothesis”}} + \Theta(n)$$

$$\leqslant \frac{2}{n} \sum_{k=\lfloor n/2 \rfloor}^{n-1} c \cdot k + \Theta(n)$$

$$= \frac{2c}{n} \underbrace{\sum_{k=\lfloor n/2 \rfloor}^{n-1} k}_{\text{“} \leqslant \frac{3}{8} n^2 \text{”}} + \Theta(n)$$

$$= c \cdot n - \Big(\underbrace{\tfrac{1}{4} cn - \Theta(n)}_{\text{“取任意大的} c / \exists c / c \text{足够大时，非负”}} \Big)$$

the fact: Random-Select has expected running time $\Theta(n)$, in the worst case $\Theta(n^2)$.

"如果最坏情况的复杂度是 $\Theta(n)$ 就好了，那就是最好的结果。毕竟所有的数字(元素)都得看一遍，因此复杂度不可能小于 $\Theta(n)$"

"怎么避免随机性呢？"

worst-case、linear-time order statistics {Blum、Floyd、Pratt、Rivest、Tarjan} in 1973

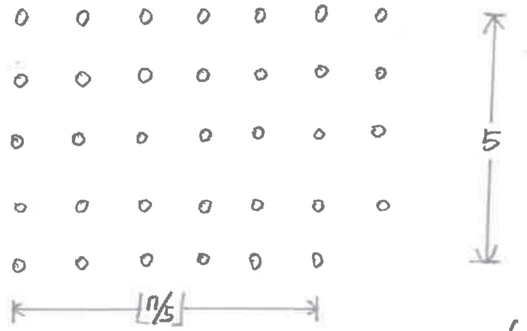
"RSA的R"

△idea: generate good pivot recursively

"找第i大的元素"

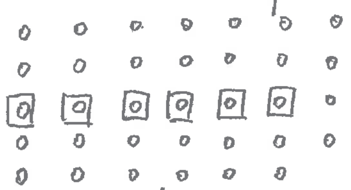
Select (i, n): "数组的大小"

① divide the n elements into $\lfloor n/5 \rfloor$ groups of 5 elements each

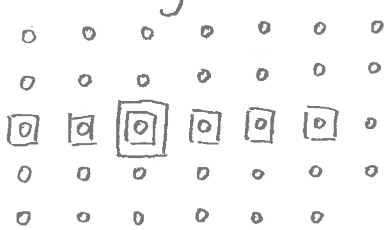


"每一列都是一组"

find the median of each group ($\Theta(n)$)



② recursively select the median x of the $\lfloor n/5 \rfloor$ group-medians ($T(n/5)$)



same as Rand-Select

③ partition with x as partition element,

let $k = \text{rank}(x)$:= "x是数组的 k-th smallest element"

④ if $i = k$ then

return x

elif $i < k$ then

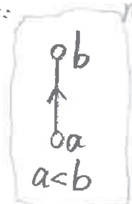
recursively select the i-th smallest element in the lower part of the array

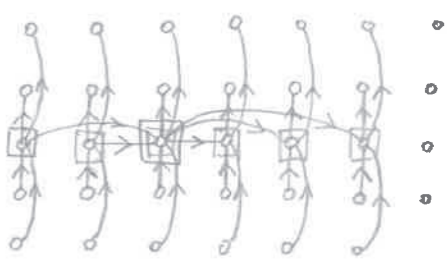
else ($i > k$) then

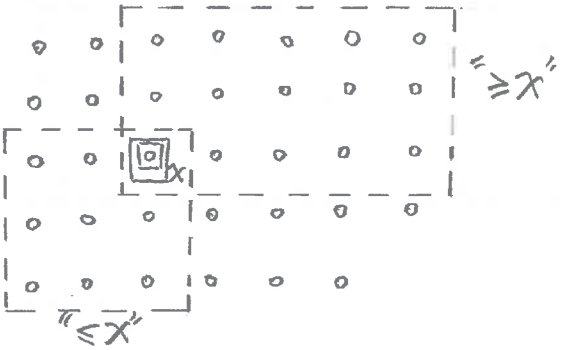
recursively select the (i-k)-th smallest element in the upper part of the array

Analysis:

notation







"已知大小关系(与x)的元素有多少个呢?"

at least (guaranteed),

$\geq 3\lfloor \lfloor n/5 \rfloor / 2 \rfloor$ elements LTE x ← "避免了极端情况的出现"

$\geq 3\lfloor \lfloor n/5 \rfloor / 2 \rfloor$ elements GTE x

(as $\lfloor \lfloor n/5 \rfloor / 2 \rfloor$ group-medians $\leq x$, as well as $\lfloor \lfloor n/5 \rfloor / 2 \rfloor$ group-medians $\geq x$)

"再根据倍数关系,得到前面的系数3"

所以,至少有大概 $\frac{3}{10}$ 的元素在 x 的左边,也至少有大概 $\frac{3}{10}$ 的元素在 x 的右边。

简化一下,

for $n \geq 50$, $3\lfloor n/10 \rfloor \geq \frac{n}{4}$ ← 最坏也只是 $\frac{1}{4} : \frac{3}{4}$ 的分法

"a powerful subroutine"!

$$T(n) \leq T(\tfrac{n}{5}) + \underbrace{T(\tfrac{7}{10}n)}_{} + \Theta(n)$$

$\frac{3}{10} : \frac{7}{10}$ 的分法

$$\approx T(\tfrac{n}{5}) + \underbrace{T(\tfrac{3}{4}n)}_{} + \Theta(n)$$

用"简化一下"部分的结论($n \geq 50$ 时, $3\lfloor n/10 \rfloor \geq \frac{n}{4}$)

claim: $T(n) \leq c \cdot n$

proof: (substitution)

assume true for smaller n

$$T(n) \leq c \cdot \tfrac{1}{5}n + c \cdot \tfrac{3}{4}n + \Theta(n)$$

$$= \tfrac{19}{20} \cdot c \cdot n + \Theta(n)$$

$$= cn - \underbrace{(\tfrac{1}{20}cn - \Theta(n))}_{\text{if } c \text{ is large enough, non-negative}}$$

$$\leq cn, \text{ for } c \text{ sufficiently large}$$

注: 5是这个算法能成功的最小数字。

3不行! 7行!