Generalità

i-esima statistica d'ordine di un insieme = i-esimo elemento più piccolo

↓

prima statistica d'ordine di un insieme = minimo
n-esima statistica d'ordine di un insieme di n elementi = massimo

Mediano di un insieme di n elementi = n° di mezzo dell'insieme

n dispari → il mediano è unico e cade all'indice i = (n+1)/2 della sequenza ordinata

n pari → due mediani, che cadono agli indici i = n/2 e i = n/2 + 1 della sequenza ordinata; tuttavia, si preferisce considerare solo il mediano di indice inferiore

Selezione (caso generale)

Input: insieme A di n numeri (distinti) e un numero i, $1 \leq i \leq n$

Output: $x \in A \mid x$ è più grande di esattamente altri $(i - 1)$ elementi di A, cioè è la i-esima statistica d'ordine di A

Possibile soluzione: ordinare i numeri con MERGE-SORT o HEAPSORT e poi indirizzare l'i-esimo elemento del vettore di uscita $\rightarrow T(n) = O(n lg n)$

Esistono però algoritmi più efficienti

Selezione di minimo e massimo

```
MINIMUM(A)
1 min ← A[1]
2 for i ← 2 to length[A]
3    do  if min > A[i]
4                then min ← A[i]
5 return min
```

O(n)

Analogamente per il massimo

Un algoritmo con (n – 1) confronti è quanto di meglio si possa fare? Sì, perché (n – 1) è il limite inferiore: si pensi a un torneo fra n elementi, dove ogni partita è un confronto fra due elementi e il minore vince; ogni elemento, eccetto il vincitore del torneo deve perdere almeno una partita → (n – 1) partite

Selezione simultanea di minimo e massimo

Possibile soluzione: calcolo indipendente di minimo e massimo $\rightarrow 2(n - 1)$ confronti

Soluzione più efficiente: si analizzano due elementi di input per volta, confrontandoli prima fra di loro e, quindi, confrontando il minore col minimo corrente e il maggiore col massimo corrente $\rightarrow$ 3 confronti per coppia = $3\lceil n/2 \rceil$ confronti in tutto

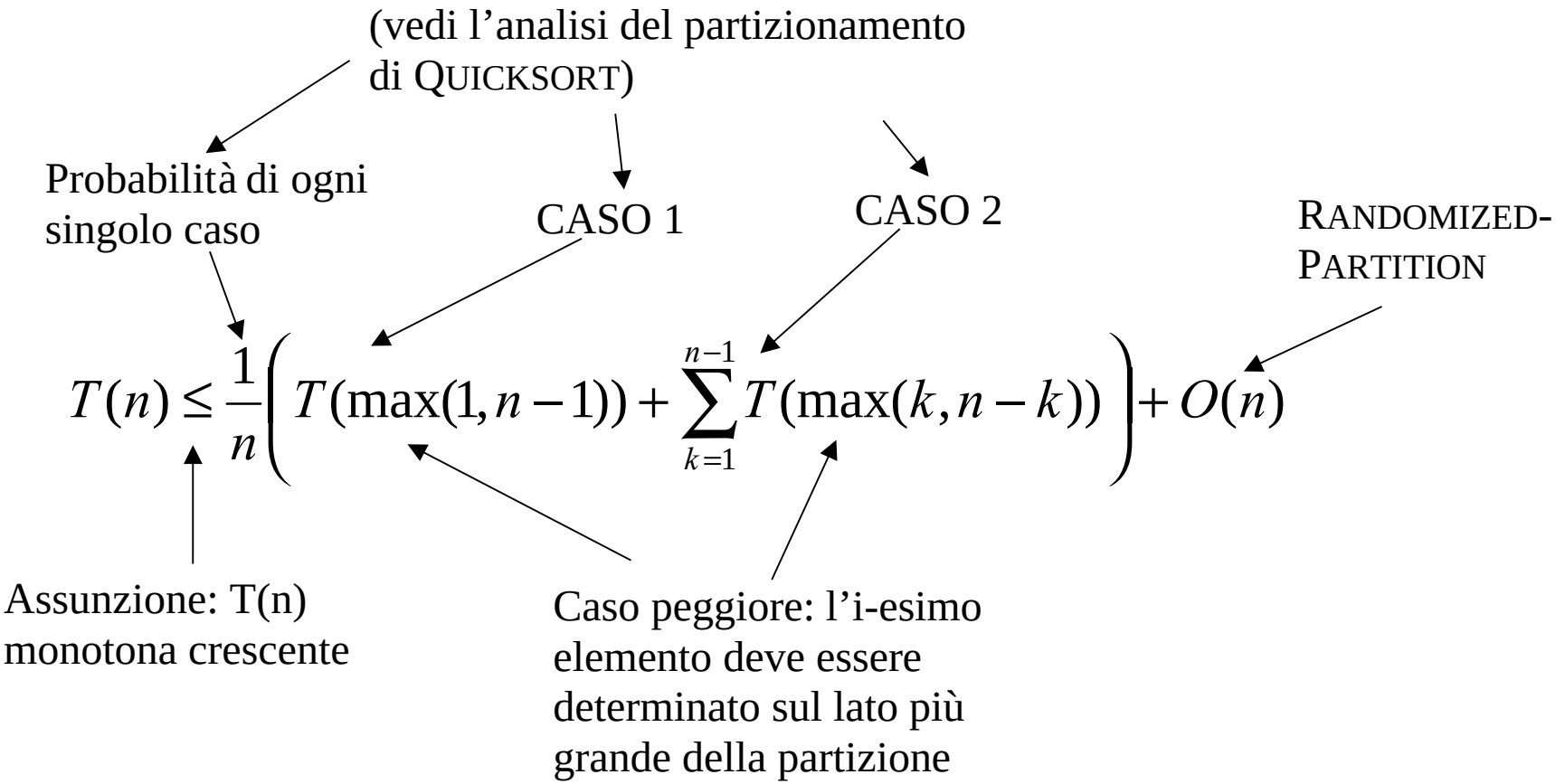
RANDOMIZED-SELECT

- Affronta il problema della selezione nel caso generale
- Come RANDOMIZED-QUICKSORT, usa RANDOMIZED-PARTITION
- Restituisce l'i-esimo elemento più piccolo di A[p .. r]

```
RANDOMIZED-SELECT(A,p,r,i)
1 if p = r
2    then return A[p]
3 q ← RANDOMIZED-PARTITION(A,p,r)
4 ▷ A[p .. r] è ora partizionato in due sottoarray non vuoti, A[p .. q] e
   A[q + 1 .. r], dove ogni elemento del primo è ≤ di ogni elemento del secondo
5 k ← q – p + 1
6 ▷ k è il numero di elementi di A[p .. q]
7 if i ≤ k
8    then return     RANDOMIZED-SELECT(A,p,q,i)
9    else return     RANDOMIZED-SELECT(A,q+1,r,i – k)
```

RANDOMIZED-SELECT: caso medio

L'algoritmo funziona bene nel caso medio e, poiché è randomizzato, nessun particolare input provoca il comportamento del caso peggiore



RANDOMIZED-SELECT: caso medio (cont.)

$$T(n) \le \frac{1}{n}\left(T(n-1) + 2\sum_{k=\lceil n/2 \rceil}^{n-1} T(k) \right) + O(n)$$

$$\max(k, n-k) = \begin{cases} k & se\ k \ge \lceil n/2 \rceil \\ n-k & se\ k < \lceil n/2 \rceil \end{cases}$$

n dispari $\rightarrow$ ogni termine T($\lceil$ n/2 $\rceil$), T($\lceil$ n/2 $\rceil$ + 1), .., T(n-1) compare due volte nella sommatoria originale

n pari $\rightarrow$ ogni termine T($\lceil$ n/2 $\rceil$ + 1), .. , T(n – 1) compare due volte nella sommatoria originale mentre T($\lceil$ n/2 $\rceil$) compare una volta

RANDOMIZED-SELECT: caso medio (cont.)

Ricorrenza da risolvere: $T(n) \le \frac{2}{n} \sum_{k=\lceil n/2 \rceil}^{n-1} T(k) + O(n)$

$$\frac{1}{n} T(n-1) + O(n) = \frac{1}{n} O(n^2) + O(n) = O(n) + O(n) = O(n)$$

nel caso peggiore (perché si potrebbe essere così sfortunati da partizionare sempre usando gli elementi più grandi rimasti)

RANDOMIZED-SELECT: caso medio (cont.)

Ipotesi: $T(n) \le cn$ per qualche costante $c > 0$
Sostituendo l'ipotesi nella ricorrenza, si ottiene $T(n) = O(n)$

$$T(n) \le \frac{2}{n} \sum_{k=\lceil n/2 \rceil}^{n-1} ck + O(n)$$

$$= \frac{2c}{n}\left(\sum_{k=1}^{n-1} k - \sum_{k=1}^{\lceil n/2 \rceil - 1} k \right) + O(n)$$

$$= \frac{2c}{n}\left(\frac{1}{2}(n-1)n - \frac{1}{2}\left(\left\lceil \frac{n}{2} \right\rceil - 1 \right)\left\lceil \frac{n}{2} \right\rceil \right) + O(n)$$

$$\le c(n-1) - \frac{c}{n}\left(\frac{n}{2} - 1 \right)\left(\frac{n}{2} \right) + O(n)$$

$$= c\left(\frac{3}{4}n - \frac{1}{2} \right) + O(n)$$

RANDOMIZED-SELECT: caso medio (cont.)

L'ipotesi è verificata se esiste una costante c > 0 tale che

$$c\left(\frac{3}{4}n-\frac{1}{2}\right)+O(n)\le cn \quad \text{cioè} \quad c\left(\frac{n}{4}+\frac{1}{2}\right)\ge O(n)$$

Dal momento che si può sempre scegliere c sufficientemente grande, l'ipotesi è dimostrata

$$\downarrow$$

$$T(n) = O(n)$$