Lez 16 marzo

Monday, March 16, 2020 10:59 AM

limiti inferiori

1) dimensione dell'input/output

2) eventi contabili

3) albero di decisioni

4) oracolo/avversario

$\boxed{\text{DOPPIO TORNEO}}$

n giocatori $\rightarrow$ primo, secondo

1) TORNEO $n-1$ confronti

2) SECONDO TORNEO $\log n - 1$ confronti

$$C(n) = n - 1 + \log n - 1 = n + \log n - 2$$

Tecnica dell'avversario per stabilire limite inferiore

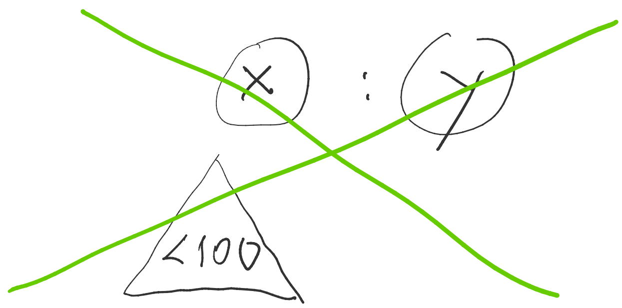
- $n-1$ confronti (per stabilire il primo) sono necessari

$$C(n) = n - 1 + X$$

j = gli incontri fatti dal campione

$M(j)$ = il numero di giocatori risultati minori del campione dopo j incontri

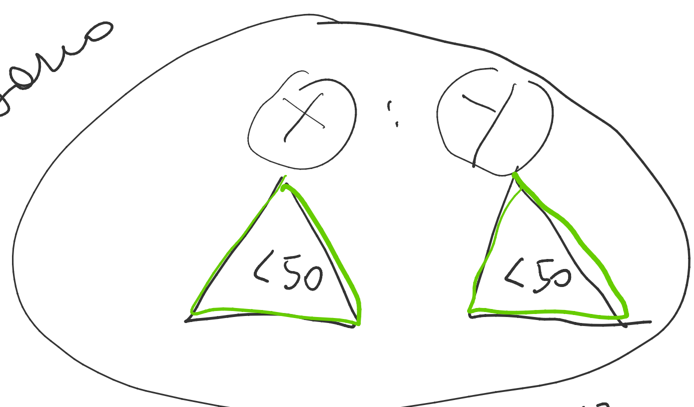
<u>ovverosia</u>: il numero di giocatori che risulti minore di un altro giocatore per transitività sia minimo



se $x > y$

se ~~$y > x$~~   <u>101</u>

Regola dell'ovverosia



$x > y$ $50 + 1$

$y > x$ $50 + 1$

$$M(j) = \begin{cases} 0 & \text{se } j = 0 \\ 2M(j-1) + 1 \end{cases}$$

$$M(j-1) = 2M(j-2) + 1$$

risolvere iterativamente

$$= 2\left(2M(j-2)+1\right)+1$$

$$= 2\left(2\left(2M(j-3)+1\right)+1\right)+1$$

$$\vdots$$

$$= 2\left(2\dots\left(2\,\underbrace{\overset{0}{\cancel{M(0)}}}\,+1\right)+1\dots\right)+1$$

$$= \cancel{2^{j}\cdot 0} + 2^{j-1} + 2^{j-2} + \dots + 2^{0} = \sum_{k=0}^{j-1} 2^{k} = 2^{j}-1$$

$$M(j) = 2^{j}-1$$

vogliamo che $n-1$ giocatori risultino minori del campione

$$2^{j}-\cancel{1} = n-\cancel{1}$$

$j = \log_2 n$ = il numero di incontri che il campione deve disputare

i perdenti del campione sono j

$$C = n-1+j-1 = \boxed{n+\log n-2}$$

DOPPIO TORNEO È OTTIMO !

QuickSort

a | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | input a

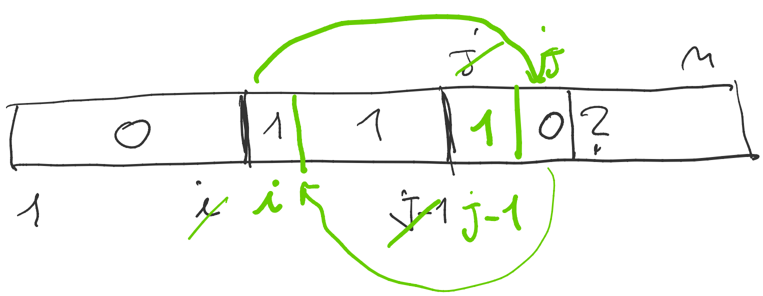
1	0	0	1	0	0	1	1	1	0	1

input

Ordina a senza usare contatori

output

0	0	0	0	0	1	1	1	1	1	1



all'iterazione generica

```
Ordina01 ( a );
i=0 ; j=1 ;
for ( j=1 ; j<=n ; j+1) {
    if (a[j]==0) {
        i++;
        scambia a[i]: a[j];
    }
    j++;
}
```

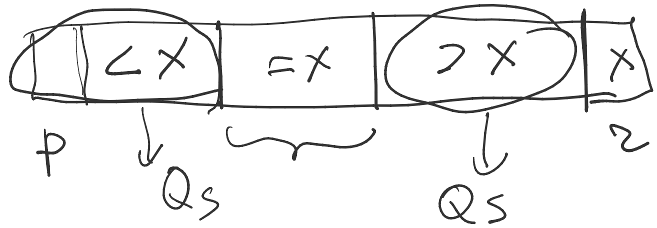
$\Theta(n)$

Partition : base di QuickSort

Problema 7.2 (Cormen pg. 153)

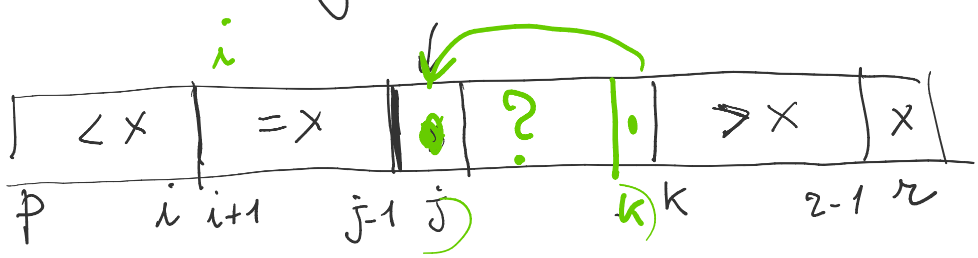
QuickSort applicato a un multinsieme

3 Partition: divide l'array in 3 parti



- QS sarà applicato ricorsivamente solo sui $< x$ e sui $> x$
- gli elementi $= x$ sono già ordinati

all'iterazione generica



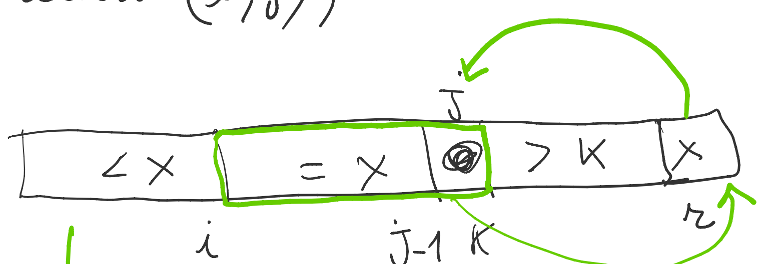
1) $a[j] = x \rightarrow j{+}{+}$

2) $a[j] < x \rightarrow i{+}{+}$ scambia $a[i]$ con $a[j]$, $j{+}{+}$

3) $a[j] > x \rightarrow k{-}{-}$ scambia $a[k]$ con $a[j]$

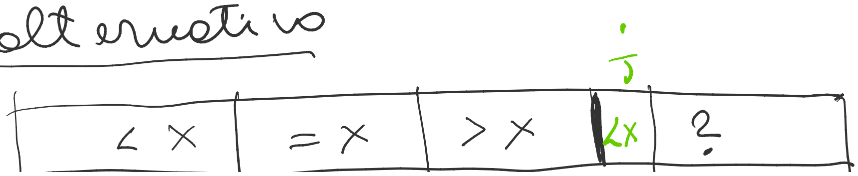
```
3Partition (a, p, r):

i = p-1; j = p; x = a[r];
while (j <= k) {

    if a[j] < x {
        i++;
        scambio a[j] con a[i];
        j++;
    }
    else if a[j] > x {
        k--;
        scambione a[j] con a[k];
    } else j++;

} scambio a[j] con a[r]  *
return (i,j);
```

$\Theta(n)$



QS(a, p, i)   QS(a, j+1, r)

Metodo alternativo



$<x$	$=x$	$>x$	$<x$	?

$<x$

3-Partition esercizio

Esercizio

input [1|2|0|1|2|1|1|2|0|0|1] el $\in \{0, 1, 2\}$

Ordinare l'array senza contatori

output [0|0|0|1|1|1|1|1|2|2|2] $\Theta(n)$

MOLTIPLICAZIONE

$P = A * B$ RAM word model

ha costo $\Theta(1)$

bit model

$A = 3856$ n

$B = 1345$ n

```
  3856 x
  1345
 ------
 17280   n op
15424    n op
  8
....
....
 ------
```

MOLTNORMALE

$\Theta(n^2)$ operazioni

$\Theta(n^2)$ operazioni

DIVIDE ET IMPERA?

$$A = A_1 * 10^{n/2} + A_2 \qquad 3856 = 38 \cdot 10^2 + 56$$

$$B = B_1 * 10^{n/2} + B_2 \qquad 1345 = 13 \cdot 10^2 + 45$$

$$A * B = (A_1 \cdot 10^{n/2} + A_2) * (B_1 \cdot 10^{n/2} + B_2) =$$

$$A_1 B_1 \cdot 10^{n} + (A_1 B_2 + A_2 B_1) * 10^{n/2} + A_2 B_2 =$$

```
MOLTRAPIDA (A, B, n):      A, B di n cifre
if (n == 1) return (A * B);
  else {
  // divide A e B in A1, B1, A2 B2 //
   X = MOLTRAPIDA (A1, B1, n/2);
   Y = MOLTRAPIDA (A2, B2, n/2);
   Z = MOLTRAPIDA (A1, B2, n/2) +
       MOLTRAPIDA (A2, B1, n/2);
   return X * 10^n + Z * 10^(n/2) + Y;
```

3