Soluzione 6: Algoritmo Quicksort

Si basa sulla partizione dell'array rispetto ad un suo elemento scelto come "pivot". L'operazione viene quindi ripetuta sulle due parti così ottenute.

1	2	3	4	5	6	7	8	9	10	11	12
0	4	2	4	3	1	5	8	6	7	9	7
					i						*j*

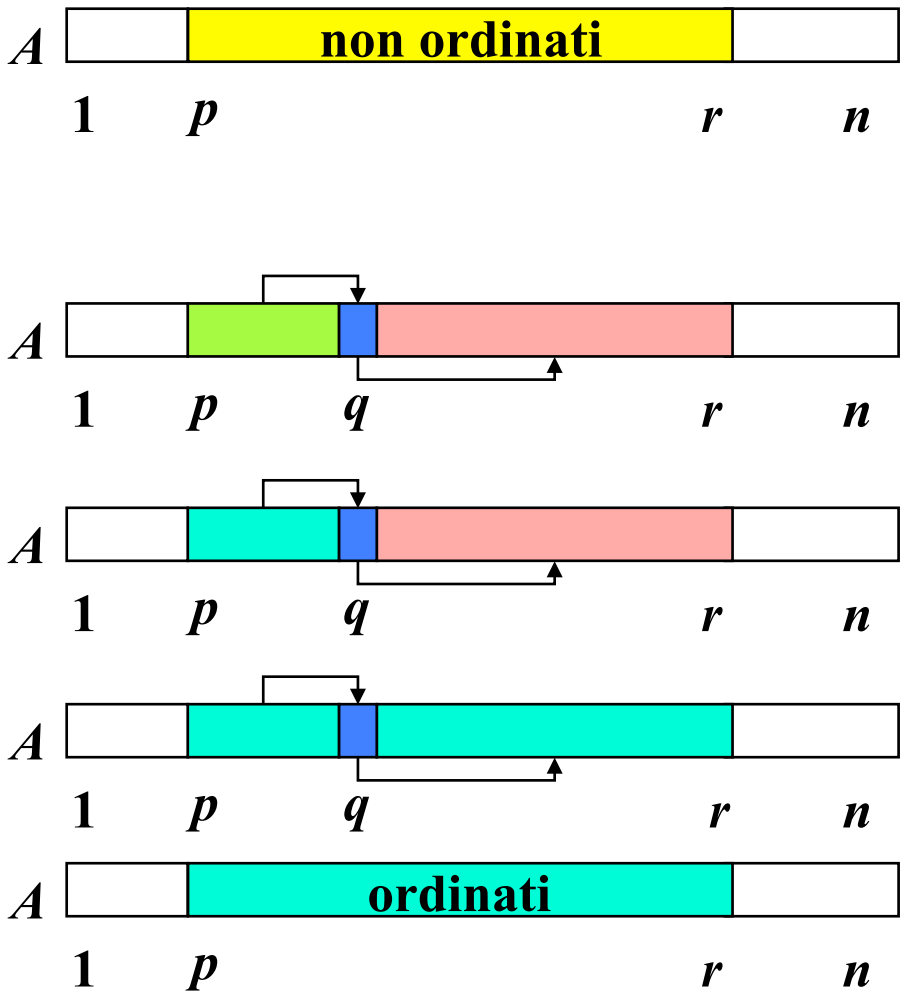
Quicksort(A,p,r)

if $p < r$

q = **Partition**(A,p,r)

Quicksort(A,p,q-1)

Quicksort(A,q+1,r)



Partition(A,p,r)

$x = A[r]$

$i = p - 1$

for $j = p$ **to** $r - 1$

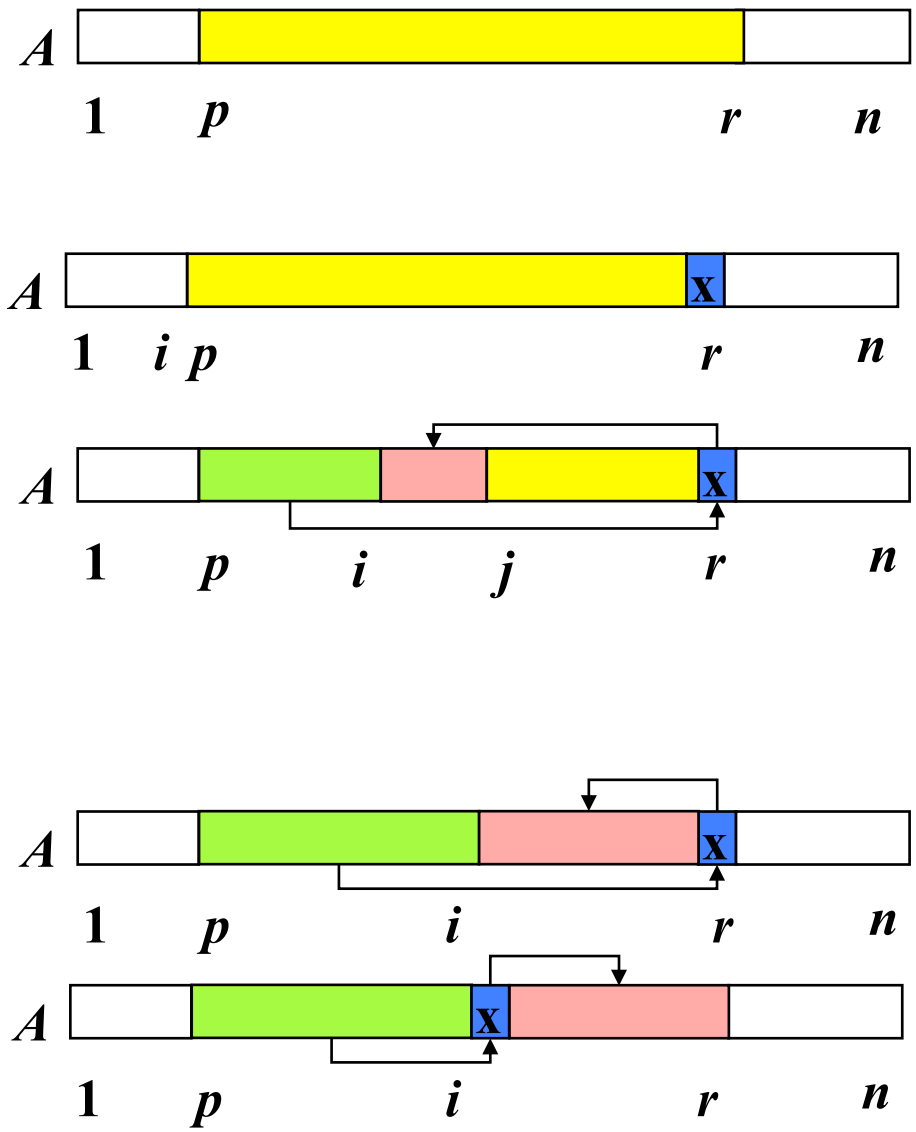
  if $A[j] < x$

    $i = i+1$

    scambia $A[i]$ e $A[j]$

scambia $A[i+1]$ e $A[r]$

return $i+1$



$$n = r - p + 1 \qquad T^P(n) = bn + a = \Theta(n)$$

```
Quicksort (A,p,r)          // Complessità massima
    if p < r
        q = Partition(A,p,r)
        Quicksort(A,p,q-1)
        Quicksort(A,q+1,r)
```

$$n = r - p + 1$$

Array ordinato o ordinato in senso inverso

$$T_{\max}^{QS}(n) = \begin{cases} c & \text{se } n \le 1 \\ bn + a + T_{\max}^{QS}(n-1) + T_{\max}^{QS}(0) & \text{se } n > 1 \end{cases}$$

$$T_{\max}^{QS}(n) = bn + a + c + T_{\max}^{QS}(n-1)$$

$$\boxed{T_{\max}^{QS}(n) = \Theta(n^2)}$$

Quicksort(*A*,*p*,*r*) // **Complessità minima**

if $p < r$ // c $\qquad n = r - p + 1$

q = **Partition**(*A*,*p*,*r*) // $T^P(n)$

Quicksort(*A*,*p*,*q*-1) // $T^{QS}_{\min}(\lceil (n-1)/2 \rceil)$

Quicksort(*A*,*q*+1,*r*) // $T^{QS}_{\min}(\lfloor (n-1)/2 \rfloor)$

$$T^{QS}_{\min}(n) \le \begin{cases} c & \text{se } n \le 1 \\ c + T^P(n) + 2T^{QS}_{\min}(\lfloor n/2 \rfloor) & \text{se } n > 1 \end{cases}$$

$$\boxed{T^{QS}_{\min}(n) = O(n \log n)}$$

Quicksort (*A*,*p*,*r*) // **Complessità media** $n = r - p + 1$

if $p < r$ **then**

q = **Partition**(*A*,*p*,*r*)

Quicksort(*A*,*p*,*q*-1)

Quicksort(*A*,*q*+1,*r*)

$$T_{med}^{QS}(n) = \begin{cases} c & \text{se } n \le 1 \\ bn + a + \frac{1}{n}\sum_{q=p}^{r}[T_{med}^{QS}(q-p) + T_{med}^{QS}(r-q)] & \text{se } n > 1 \end{cases}$$

$$= \begin{cases} c & \text{se } n \le 1 \\ bn + a + \frac{1}{n}\sum_{j=0}^{n-1} T_{med}^{QS}(j) + \frac{1}{n}\sum_{j=0}^{n-1} T_{med}^{QS}(j) & \text{se } n > 1 \end{cases}$$

$$= \begin{cases} c & \text{se } n \le 1 \\ bn + a + \frac{2}{n}\sum_{j=0}^{n-1} T_{med}^{QS}(j) & \text{se } n > 1 \end{cases}$$

$$\boxed{T_{med}^{QS}(n) = O(n \log n)}$$

Per $n > 1$ $\quad T_{med}^{QS}(n) = bn + a + \frac{2}{n}\sum_{i=0}^{n-1} T_{med}^{QS}(i)$

e moltiplicando per n otteniamo

$$nT_{med}^{QS}(n) = bn^2 + an + 2\sum_{j=0}^{n-1} T_{med}^{QS}(j)$$

Per $n = 2$ $\quad T_{med}^{QS}(2) = 2b + a + 2c$

Per $n > 2$ $\quad nT_{med}^{QS}(n) - (n-1)T_{med}^{QS}(n-1)$

$$= 2T_{med}^{QS}(n-1) + (2n-1)b + a$$

e dunque

$$nT_{med}^{QS}(n) = (n+1)T_{med}^{QS}(n-1) + (2n-1)b + a$$

$$nT_{med}^{QS}(n) = (n+1)T_{med}^{QS}(n-1) + (2n-1)b + a$$

dividendo per $n(n+1)$

$$\frac{1}{n+1}T_{med}^{QS}(n) = \frac{1}{n}T_{med}^{QS}(n-1) + \frac{(2n-1)b+a}{n(n+1)}$$

ponendo $f(n) = \dfrac{1}{n+1}T_{med}^{QS}(n)$ otteniamo

$$f(n) = f(n-1) + \frac{(2n-1)b+a}{n(n+1)} \quad \text{per } n > 2$$

$$f(2) = (2b + a + 2c)/3 = c_1$$

$$f(2) = (2b + a + 2c)/3 = c_1$$

$$f(n) = f(n-1) + \frac{(2n-1)b + a}{n(n+1)} \quad \text{per } n > 2$$

la cui soluzione è

$$f(n) = f(2) + \sum_{j=3}^{n} \frac{(2j-1)b + a}{j(j+1)}$$

$$\leq c_1 + \sum_{j=3}^{n} \frac{(2j+2)(b+a)}{j(j+1)}$$

$$= c_1 + c_2 \sum_{j=3}^{n} \frac{1}{j}$$

$$\leq c_1 + c_2 \int_2^n \frac{1}{x} dx = c_1 + c_2 (\ln n - \ln 2)$$

Infine

$$
\begin{aligned}
T_{med}^{QS}(n) &= (n+1)\,f(n) \\
&\le (n+1)\left[c_1 + c_2(\ln n - \ln 2)\right] \\
&= O(n \log n)
\end{aligned}
$$

Quindi $\boxed{T_{med}^{QS}(n) = O(n \log n)}$

La complessità media O(n log n) di Quick-Sort vale soltanto se tutte le permutazioni dell'array in ingresso sono ugualmente probabili.

In molte applicazioni pratiche questo non è vero!!!

Vi sono applicazioni in cui le permutazioni quasi ordinate sono molto più probabili e questo può aumentare la complessità media fino ad $O(n^2)$.

```
Randomized-Partition(A,p,r)
    i = Random(p,r)
    scambia A[i] e A[r]
    return Partition(A,p,r)
```

```
Randomized-Quicksort(A,p,r)
    if p < r then
        q = Randomized-Partition(A,p,r)
        Randomized-Quicksort(A,p,q-1)
        Randomized-Quicksort(A,q+1,r)
```