Algoritmi e Strutture Dati (21/10/2021)

ESERCIZIO : / DOMANDE

① $f(n) = O(g(n))$ sse $g(n) = \Omega(f(n))$

$(\Rightarrow)$ sia $f(n) = O(g(n))$ ovvero esiste $c > 0$, esiste n_0 $\forall n \geq n_0$

$$\boxed{0 \leq f(n) \leq c\, g(n)} \quad (*)$$

vogliamo dimostrare $g(n) = \Omega(f(n))$ cioè esiste $d > 0$, n_1 t.c. $\forall n \geq n_1$

$$\boxed{0 \leq d\, f(n) \leq g(n)}$$

partendo da $(*)$ dividendo per $c > 0$ $\quad \forall n \geq n_0$

$$0 \leq \frac{1}{c} f(n) \leq g(n)$$

abbiamo finito : $d = \frac{1}{c} > 0$ e $n_1 = n_0$

$(\Leftarrow)$ analogo

② $\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$

$\subseteq$

$\supseteq$

$(\subseteq)$ sia $f(n) = \Theta(g(n))$ $\quad$ vogliamo $\quad$ $f(n) = O(g(n))$
$\Omega(g(n))$

$\Rightarrow f(n) \in O(g(n)) \cap \Omega(g(n))$

$\exists\, c, d > 0 \quad \exists\, n_0 \quad \forall n \geq n_0$

$$0 \leq c\, g(n) \leq f(n) \leq d\, g(n)$$

vorremmo

$f(n) = O(g(n))$ $\qquad$ $0 \leq f(n) \leq d_1\, g(n) \quad \forall n \geq n_1$

si può prendere $d_1 = d$ $\quad n_1 = n_0$

$f(n) = \Omega(g(n))$ $\qquad$ $0 \leq c_1\, g(n) \leq f(n) \quad \forall n \geq n_2$

si può prendere $c_1 = c$, $n_2 = n_0$

(c) $\underbrace{f(n) = O(g(n))}$ e $\underbrace{f(n) = \Omega(g(n))}$ ⟹ $f(n) = \Theta(g(n))$

$\exists d > 0, n_0 \ \forall n \geq n_0$ $\qquad$ $\exists c > 0, n_1 \ \forall n \geq n_1$

$0 \leq f(n) \leq d\, g(n)$ $\qquad$ $0 \leq c\, g(n) \leq f(n)$

vorremmo :

$$0 \leq c\, g(n) \leq f(n) \leq d\, g(n) \qquad \forall n \geq \max\{n_0, n_1\}$$

ESERCIZIO : RICORRENZE

(a) $T(n) = 2T\left(\frac{n}{2}\right) + \log n$

$\uparrow a=2$ $\quad \uparrow b=2$ $\quad \uparrow f(n)$

$f(n) = \log n$ $\qquad$ $n^{\log_b a} = n^{\log_2 2} = n$

$f(n) = O(n^{\log_b a}) = O(n)$ [possibile caso 1]

voglio $f(n) = O(n^{(\log_b a) - \varepsilon}) = O(n^{1-\varepsilon})$

$\varepsilon > 0$ $\qquad$ $0 < \varepsilon < 1$

$$\frac{\log n}{n^{1-\varepsilon}} \xrightarrow[\substack{\varepsilon < 1 \\ n \to \infty}]{} 0$$

⟹ Master Theorem

$$T(n) = \Theta(n^{\log_b a}) = \Theta(n)$$

(b) $T(n) = 2T\left(\frac{n}{2}\right) + n^2$

$2 = a$, $\quad 2 = b$, $\quad \uparrow$

$n^{\log_b a} = n$ $\qquad$ $f(n) = n^2 = \Omega(n^{\log_b a + \varepsilon})$

$= \Omega(n^{1+\varepsilon})$ $\qquad$ $0 < \varepsilon < 1$

possibile caso 3 ?

regolarità $\quad a\, f\left(\frac{n}{b}\right) \leq k\, f(n) \qquad 0 < k < 1$ opportuno

$$2\, f\left(\frac{n}{2}\right) \leq k\, f(n)$$

$$2 \left(\frac{n}{2}\right)^2 \leq k\, n^2$$

$$2\,\frac{n^2}{4} = \frac{1}{2} n^2 \leq k\, n^2 \qquad \text{ok per } k = \frac{1}{2}$$

$\cancel{k = 1/2}$

$\hookrightarrow T(n) = \Theta(f(n)) = \Theta(n^2)$

c) $T(n) = 2T\left(\frac{n}{2}\right) + \underline{n \log n}$

$n^{\log_b a} = n \qquad f(n) = n \log n$

$f(n) = n \log n = \Omega(n^{\log_b a}) = \Omega(n)$ $\qquad$ CASO 3?

$\exists\, \varepsilon > 0$

$$f(n) = n \log n = \Omega(n^{\log_b a + \varepsilon}) = \Omega(n^{1+\varepsilon}) \qquad \varepsilon > 0$$

$$\frac{f(n)}{n^{\log_b a + \varepsilon}} = \frac{n \log n}{n^{1+\varepsilon}} = \frac{\log n}{n^{\varepsilon}} \xrightarrow[n \to \infty]{} 0 \qquad \forall \varepsilon > 0$$

$\log_2 n$

$n \log n \qquad\qquad n \log n \qquad 0$

$\frac{n}{2}\log\frac{n}{2} \quad \frac{n}{2}\log\frac{n}{2} \qquad\qquad n \log\frac{n}{2}$

$\frac{n}{4}\log\frac{n}{4} \quad \frac{n}{4}\log\frac{n}{4} \quad \frac{n}{4}\log\frac{n}{4} \quad \frac{n}{4}\log\frac{n}{4} \qquad n\log\frac{n}{4}$

$\log n - 1$

$c \quad c \quad c \quad c \quad c \quad c$

$\underbrace{\qquad\qquad\qquad}_{nc}$

$\log n$

$T(n) = 2T\left(\frac{n}{2}\right) + n \log n$

$$\left(\sum_{j=0}^{\log n - 1} n \log \frac{n}{2^j}\right) + cn$$

$$\underbrace{\left(\sum_{J=0}^{\log n-1} n \log \frac{n}{2^J}\right)}_{\text{}} + cn$$

$$\parallel$$

$$n \sum_{J=0}^{\log n-1} \left(\log n - \log 2^J\right) + cn$$

$$n \left(\underbrace{(\log n)^2}_{\sum_{J=0}^{\log n-1} \log n} - \sum_{J=0}^{\log n-1} \underbrace{\log 2^J}_{J \log 2} \right) + cn$$

$$n\left((\log n)^2 - \log 2 \underbrace{\sum_{J=0}^{\log n-1} J}_{\frac{(\log n)(\log n-1)}{2}}\right) + cn \sim n (\log n)^2$$

$$\frac{(\log n)(\log n-1)}{2} = \frac{(\log n)^2 - \log n}{2}$$

$$\boxed{T(n) = \Theta\left(n (\log n)^2\right)}$$ IPOTESI

① $T(n) = O\left(n (\log n)^2\right) \leftarrow$

② $T(n) = \Omega\left(n (\log n)^2\right) \leftarrow$

<u>vediamo 2</u> :

$$\boxed{T(n) \geq d\, n (\log n)^2}$$

esiste $d > 0$
per n "grande"

$$T(n) = 2T\left(\frac{n}{2}\right) + n \log n$$

$\frac{n}{2} < n \Rightarrow$ per ip. ind.

$$\boxed{T\left(\frac{n}{2}\right) \geq d \frac{n}{2} \left(\log \frac{n}{2}\right)^2}$$

$$\geq \cancel{2} d \frac{n}{\cancel{2}} \left(\log \frac{n}{2}\right)^2 + n \log n$$

$$= dn \left(\log n - \log 2\right)^2 + n \log n$$

$$= \underline{dn} \left((\log n)^2 - 2 \log 2 \log n + \underline{(\log 2)^2}\right) + \underline{n \log n}$$

$$= dn(\log n)^2 + n\log n - 2d\log 2\, n\log n + dn(\log 2)^2$$

$$= dn(\log n)^2 + n\Big[\underbrace{\log n}_{\geq 0}\,\boxed{(1-2d\log 2)}_{\geq 0} + \underbrace{d(\log 2)^2}_{ok}\Big)$$

ok

? $\geq dn(\log n)^2$

$$1 - 2d\log 2 \geq 0$$

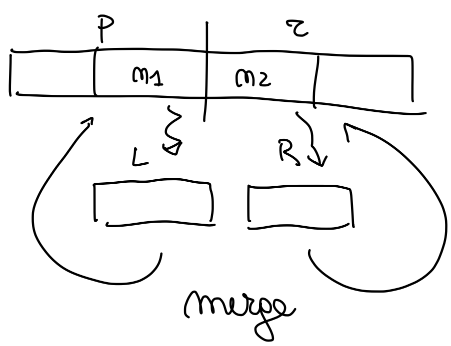
$$0 \leq d \leq \frac{1}{2\log 2}$$

OK

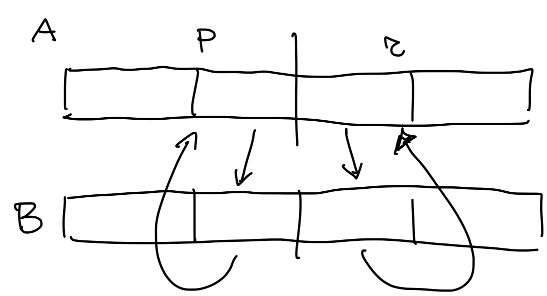
(1) idem

$$\boxed{T(n) = T\left(\frac{n}{a}\right) + T\left(\frac{n}{b}\right) + f(n)}$$

* MERGESORT "VARIANTI"



MergeSort (A)
 n = A.length
 B[1...n]
 MergeSortRic(A, B, 1, n)



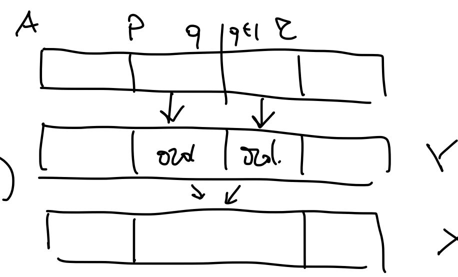
MergeSortAlt (A, B, p, r, dest A)

dest A
true ⇒ destinazione è A
false destinazione è B

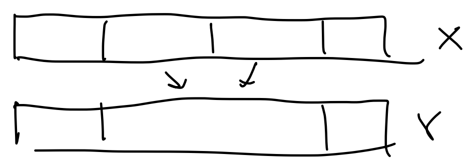
```
MergeSort (A)
    n = A.length
    B[1...n]
    MergeSortAlt (A, B, 1, n, true)

MergeSortAlt (A, B, p, r, dest A)
    if p < r
        q = ⌊(p+r)/2⌋

        MergeSortAlt (A, B, p, q, not dest A)
        MergeSortAlt (A, B, q+1, r, not dest A)

        if dest A
            Merge (B, A, p, q, r)
        else
            Merge (A, B, p, q, r)
    else
        if not dest A
            B[p] = A[p]
```

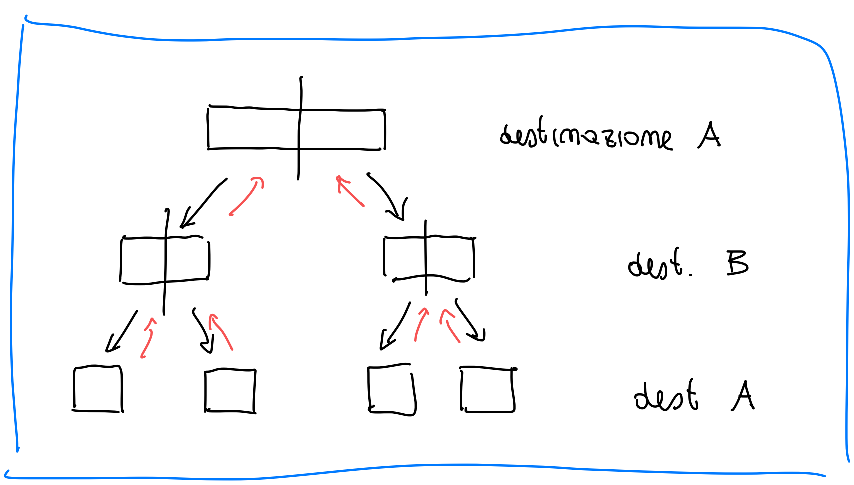
destinazione A oppure B ⇝ X

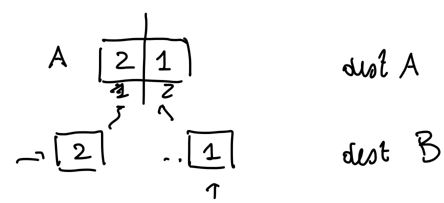
e quindi chiamo Y altro array



Merge (X, Y, p, q, r)

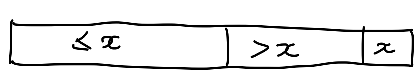






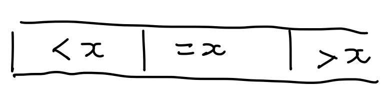
* Quick Sort con tripartizione

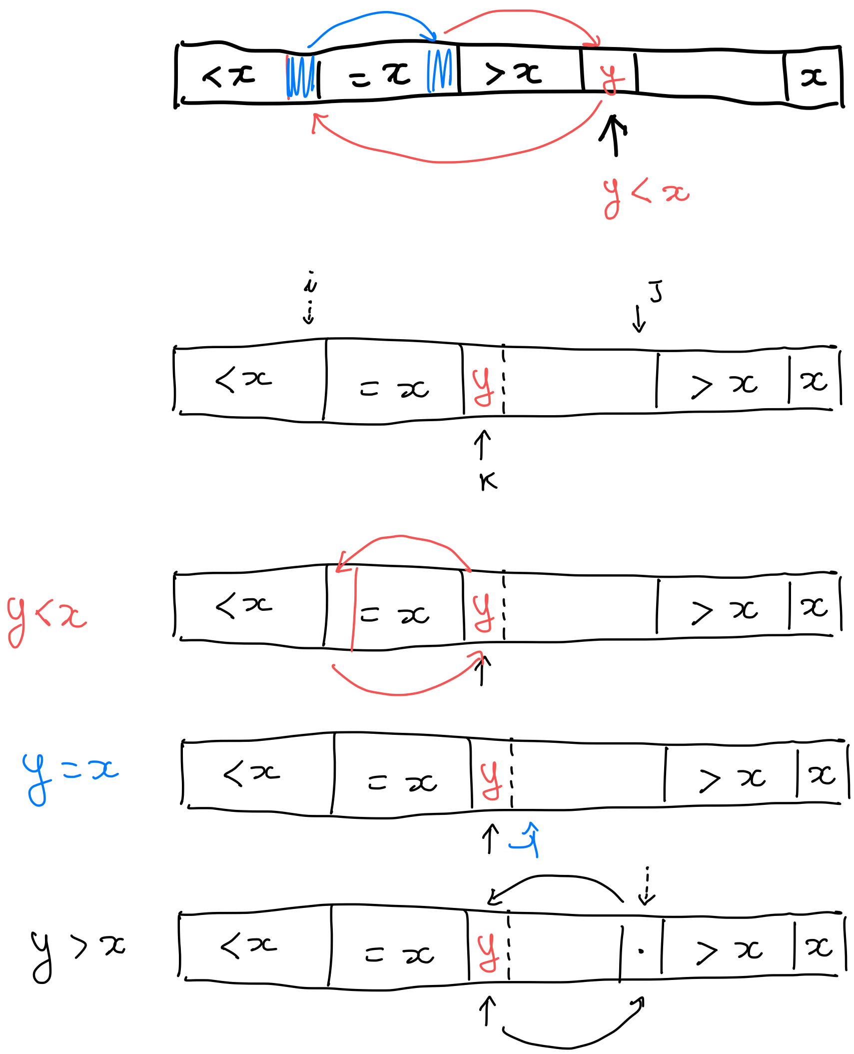
problema:



duplicati

tripartizione





< x
= x
> x
y
x
y < x
i
j
< x
= x
y
> x
x
k
y < x
< x
= x
y
> x
x
y = x
< x
= x
y
> x
x
y > x
< x
= x
y
> x
x