Lezione 16 aprile

Thursday, April 16, 2020 4:04 PM

ABR Operazioni del dizionario

- ricerca
- inserzione
- cancellazione
- min, max
- predecessore e successore

$O(h)$

h = altezza dell'albero

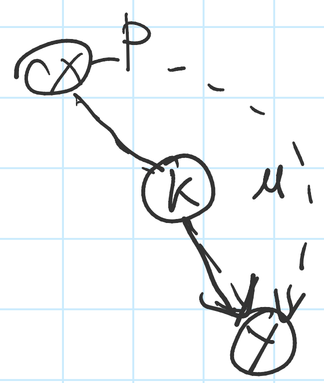
Ordinamento $\equiv$ Visita in ordine simmetrico $\Theta(n)$

Cancellazione di una chiave K

Ricerca di K

2 casi: 1) K ha un puntatore = NULL

2) K ha i 2 puntatori $\neq$ NULL



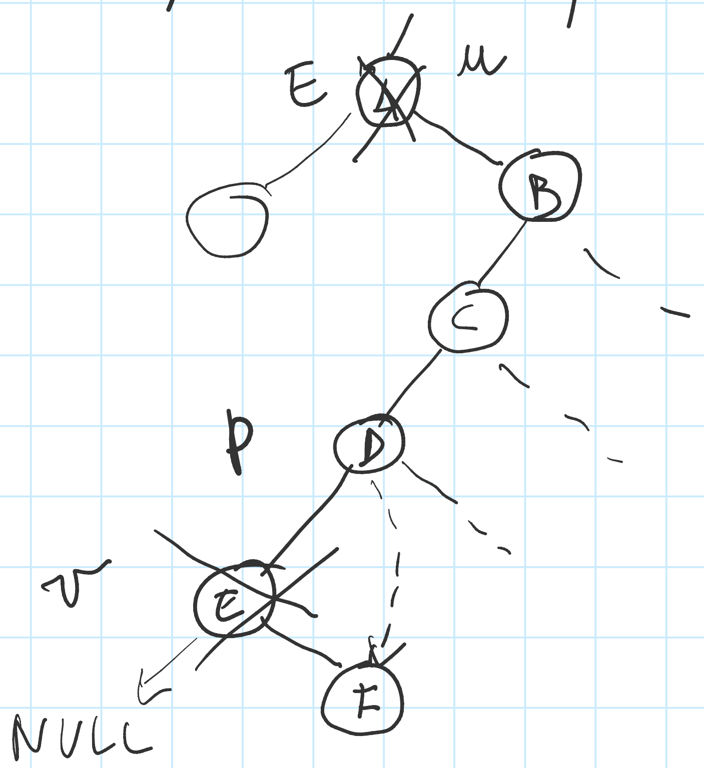
1) a) $u.sx = \emptyset$

```
if p.chiave < u.chiave
    p.dx = u.dx
else
    p.sx = u.dx
```

b) $u.dx = \emptyset$

simmetrico

2) $u.sx \neq \emptyset$ e $u.dx \neq \emptyset$



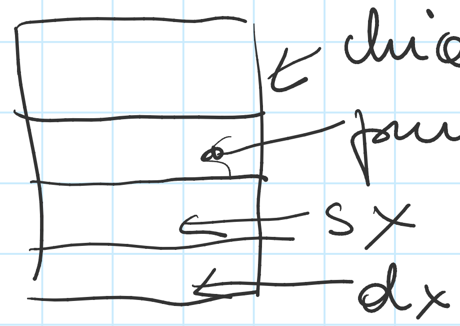
eliminiamo SUCCESSORE

(siamo nel caso 1.b)

Scambio (u.chiave con v.chiave)

CORMEN: TRASPLANT (molto complicato!)

In ogni modo: elemento (chiave, dati satelliti)

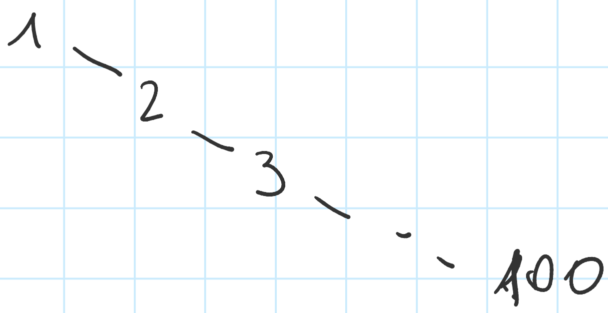


$$\lceil \log n \rceil \leq h \leq n$$

albero quasi bilanciato

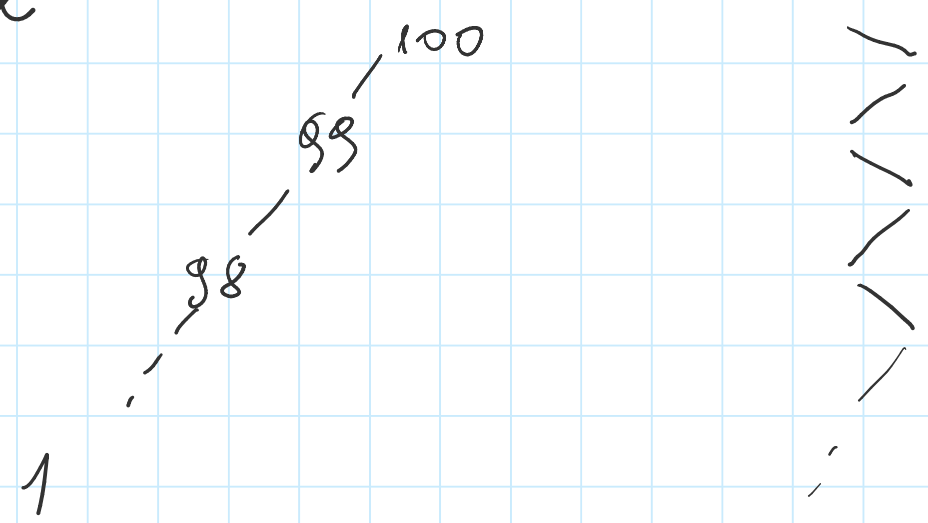
albero degenere

se i dati in ordine crescente



$h = 99$!!

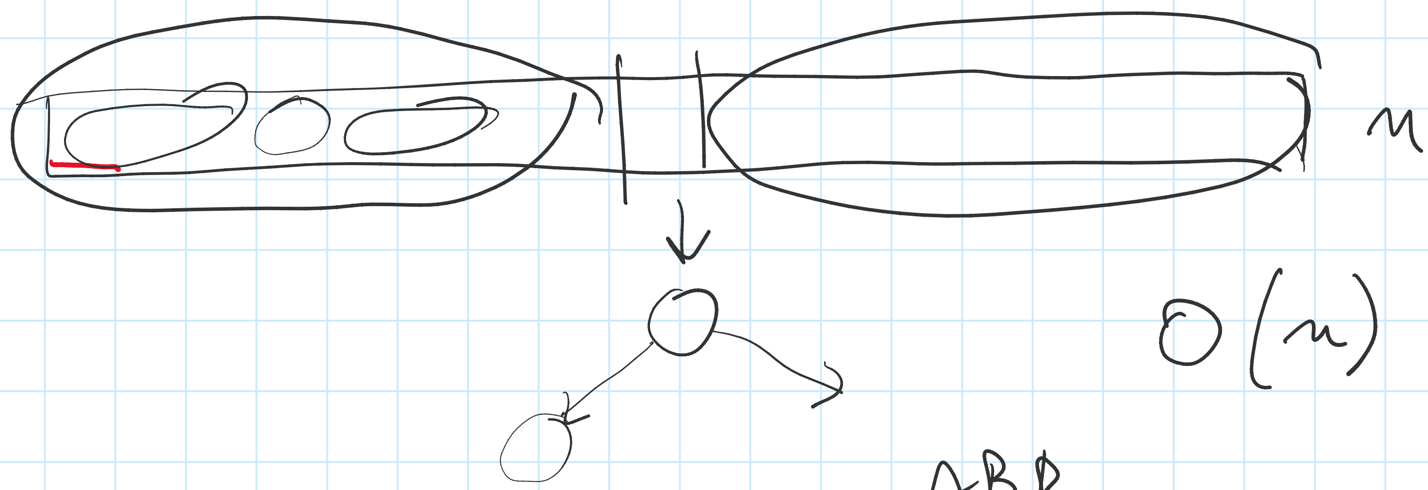
ricerca nell'ABR (caso pessimo) ≡ ricerca sequenziale



ogni inserzione aumenta l'altezza di 1 ≡ ABR caso pessimo

$h = O(n)$

IN MEDIA $h = O(\log n)$



$O(n)$

Costruzione alb. binario bilanciato ABR da array ordinato in $\Theta(n)$.

Garantire una complessità logaritmica anche nel caso pessimo.

- Si permette all'ABR di sbilanciarsi "un po'"

Mantenere l'albero completamente bilanciato in modo semplice? **NO** costo $O(n)$ per operazione.

Alberi ABR bilanciati in altezza

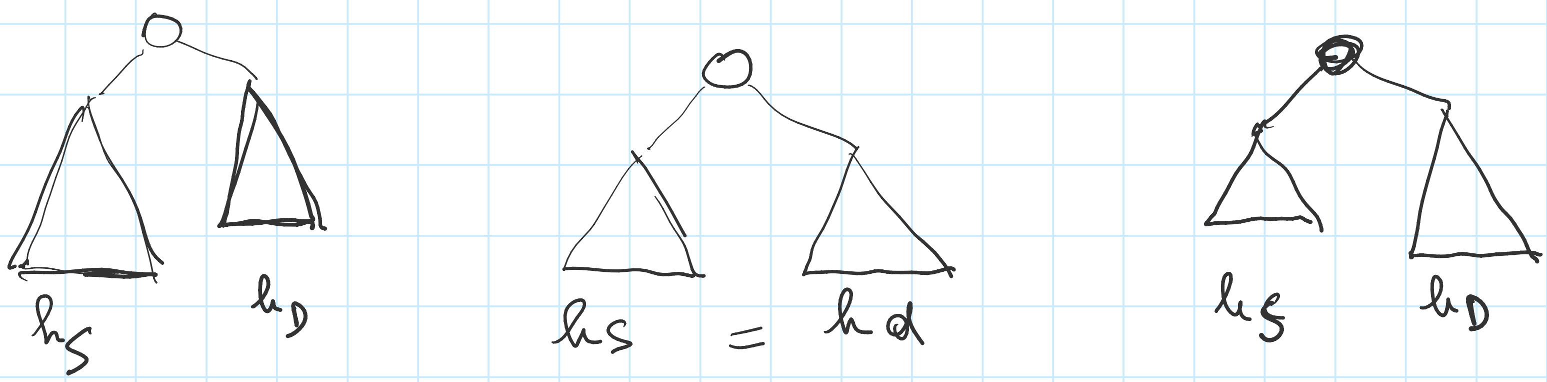
- Alberi 2-3 $\longleftrightarrow$ B-alberi
- Alberi rosso-neri $\longleftrightarrow$ Cormen... Rivest
- Alberi AVL

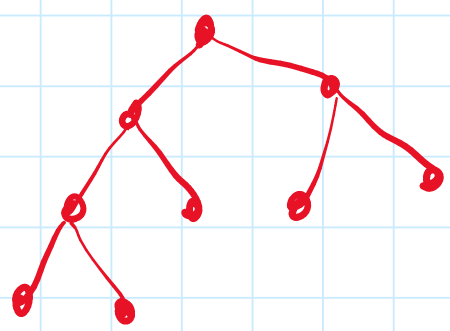
h_S altezza del sottoalb. sinistro

h_D altezza del sottoalbero destro

Albero **AVL** è un albero binario di ricerca, tale che per ogni nodo:

$$|h_S - h_D| \le 1$$





AVL

Teo: Un albero AVL ha altezza $O(\log n)$

Prova: Studiamo gli alberi più sbilanciati possibile.

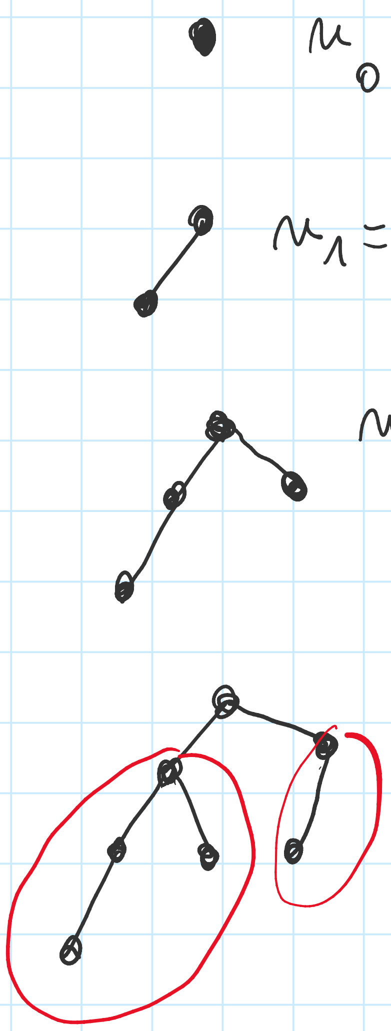
Studiamo questi alberi per altezza fissata:

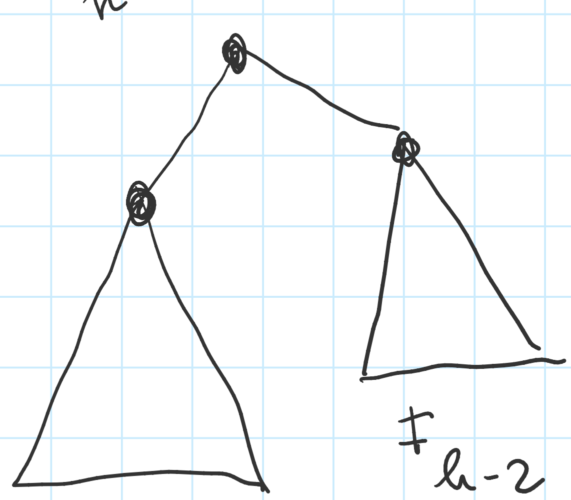
$h=0$ F_0 $n_0=1$

$h=1$ F_1 $n_1=2$

$h=2$ F_2 $n_2=4$

$h=3$ F_3 $n_3=7$





Alberi di Fibonacci, sono alb. AVL che raggiungono altezza h e sono più sbilanciati possibile.

n_h = numero di nodi di F_h

$$\boxed{n_h = n_{h-1} + n_{h-2} + 1}$$

f_h = h-esimo numero di Fibonacci

$$\boxed{f_h = f_{h-1} + f_{h-2}}$$

h	0	1	2	3	4	5	6	7	8
n_h	1	2	4	7	12	20	33	54	88
f_h	0	1	1	2	3	5	8	13	21

Teo: $n_h = f_{h+3} - 1$. Prova: induzione

base $h = 0$ $\quad n_h = 2 - 1 = 1$ vero

ipotesi induttiva

$$n_h = \underbrace{f_{h+2} - 1 + f_{h+1}}_{f_{h+3}} \cancel{-1} + \cancel{1}$$

$$n_h = f_{h+3} - 1 \qquad \text{vero}$$

$$f_h = \frac{\phi^h - (1-\phi)^h}{\sqrt{5}} \quad \text{con } \phi = \frac{1+\sqrt{5}}{2}$$

$$f_h \geq c^h, \quad c > 1 \text{ e } h > 2$$

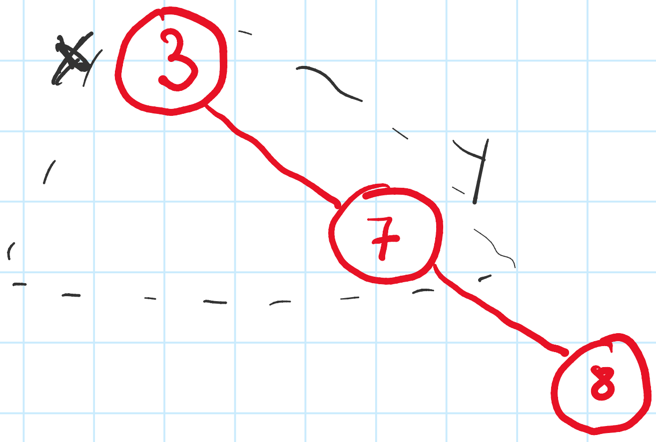
$$n_h = f_{h+3} - 1 \geq c^h \qquad n \geq n_h$$

$$n \geq c^h \qquad h \leq O(\log n)$$

$$h \leq 1.4 \log n + c_1$$

Come si mantiene la proprietà?

Esempio: AVL su chiavi $\{3, 7, 8, 4, 2, 5, 1\}$



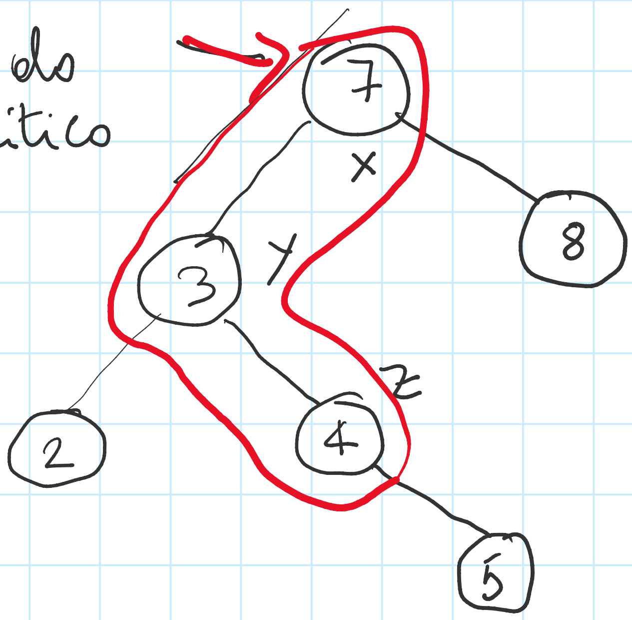
proprietà violata

⇓

<u>rotazione</u> semplice sinistra

h=1

nodo critico

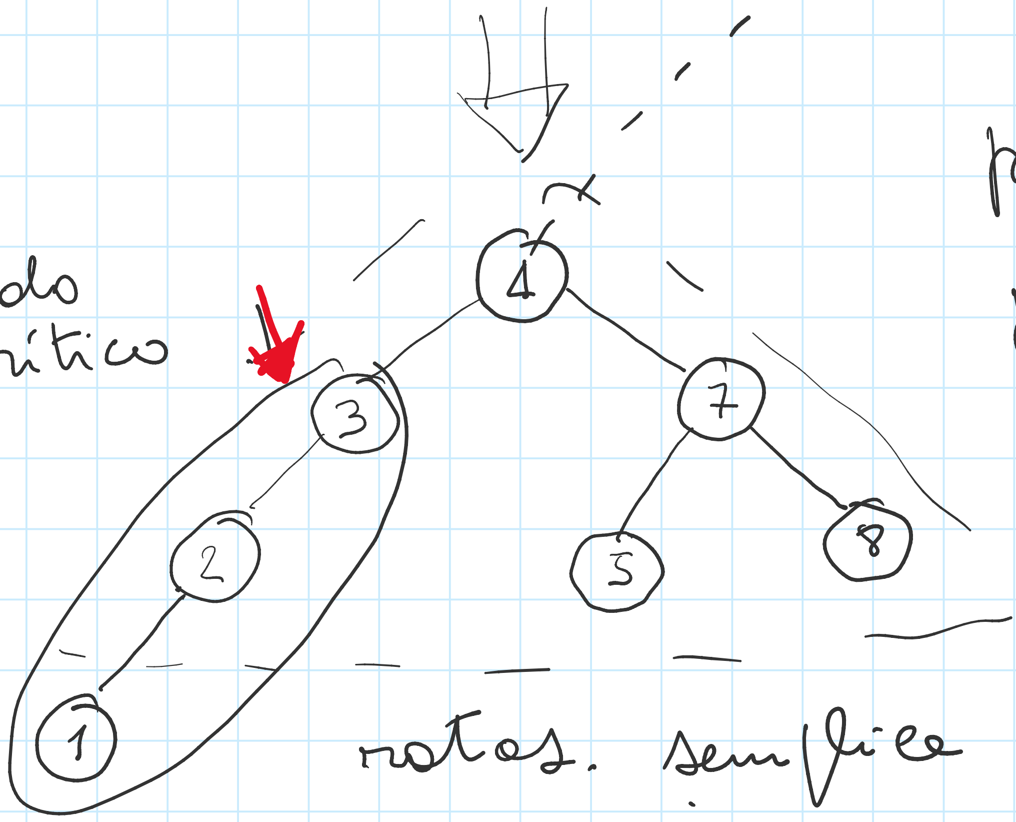


- proprietà di ABR rispettate
-

<u>rotazione</u> doppia destra

⇓

Nodo critico

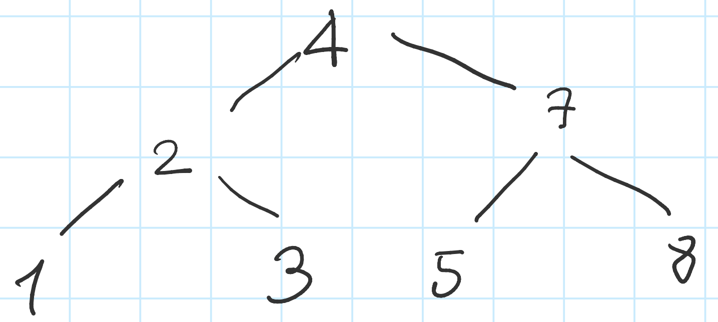


proprietà ABR

proprietà altezza

h = 2

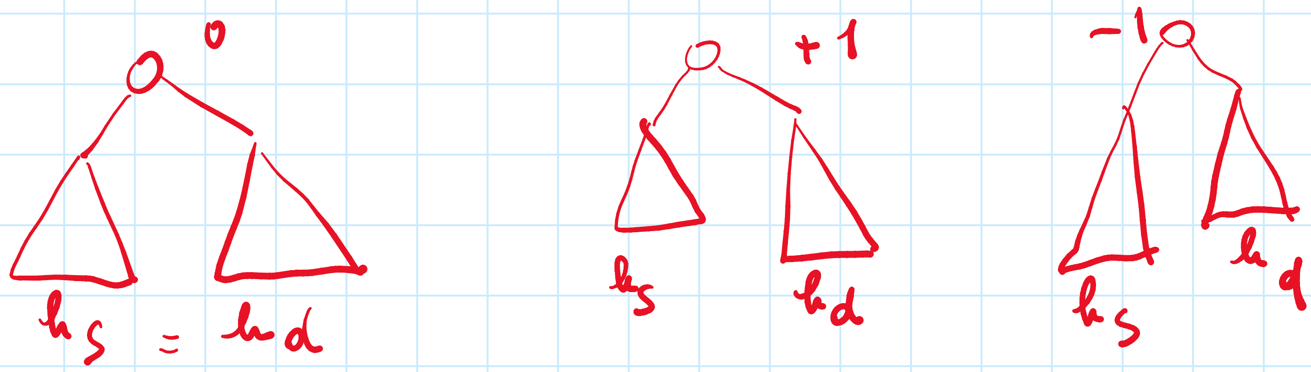
rotaz. semplice destra



Ogni nodo memorizza un **fattore di bilancio**

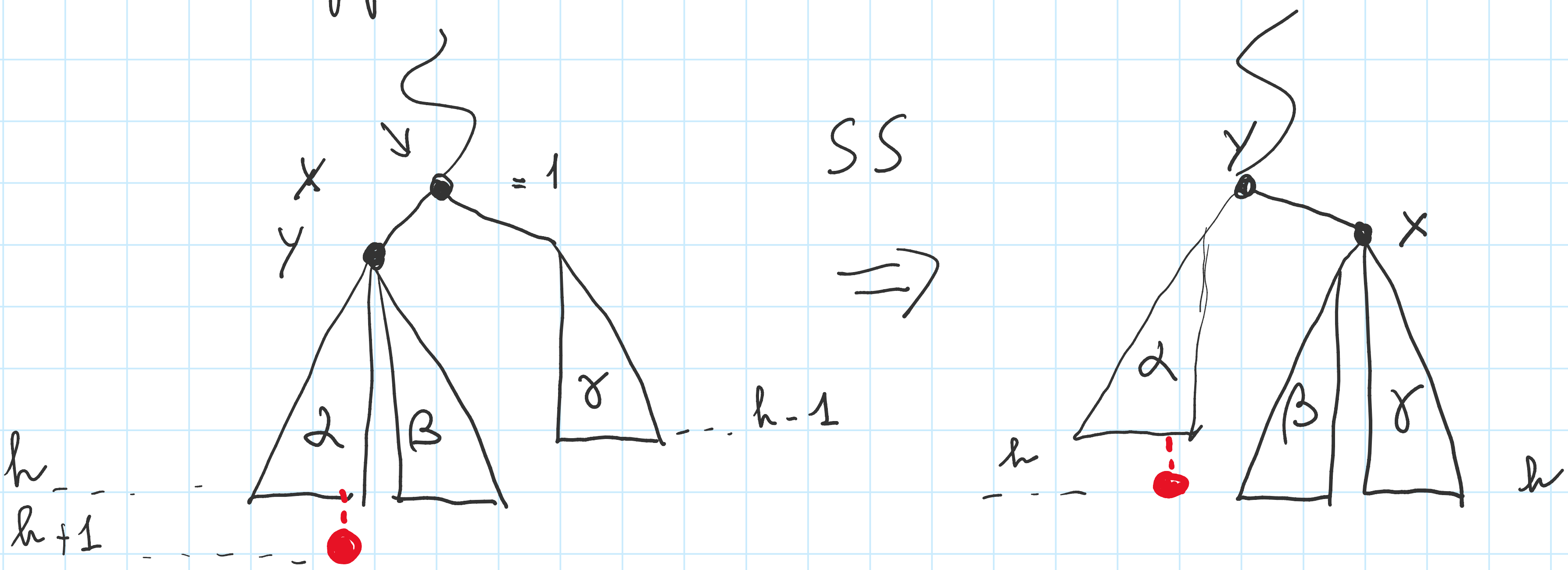
può avere 3 valori

-1 $\quad$ $=0$ $\quad$ $+1$



Rotazioni

- semplice a sinistra
- doppia



proprietà ABR è mantenuta

altezza di prima dell'inserzione ripristinata

operazione locale al sottoalbero del nodo critico.

complessità $\Theta(1)$ puntatori
inserzione $O(\log n)$ in AVL.

Rotazione doppia