Algoritmi e Strutture Dati

Lezione 28

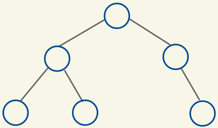
26 novembre 2025

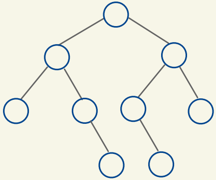
Alberi bilanciati

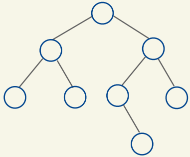
Alberi perfettamente bilanciati

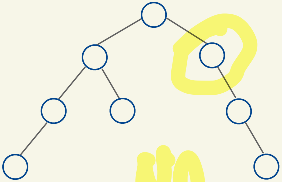
Definizione

Un albero binario è detto *perfettamente bilanciato* quando *per ogni nodo* la differenza in valore assoluto tra i numeri di nodi presenti nei suoi sottoalberi sinistro e destro è al massimo 1

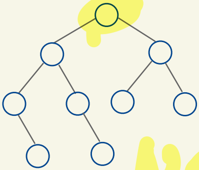






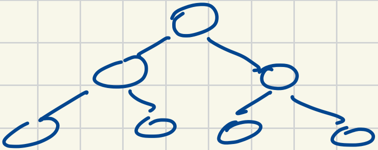


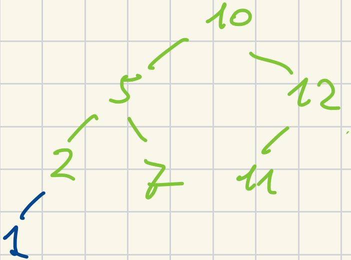
NO

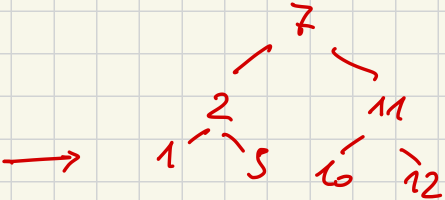


NO

Perf bilanciat. $\longrightarrow$ $h = \lfloor \log_2 n \rfloor$





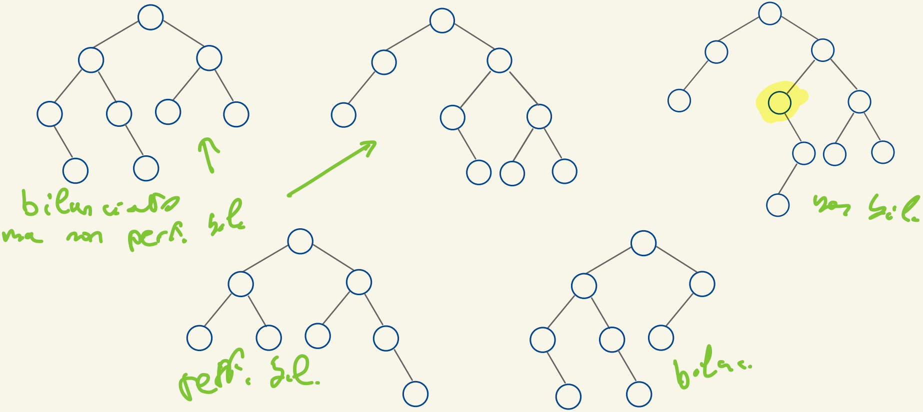


inserimento molto costoso!

Alberi bilanciati in altezza

Definizione

Un albero binario è detto *bilanciato* (in altezza) o AVL[1] quando *per ogni nodo* la differenza in valore assoluto tra le altezze dei suoi sottoalberi sinistro e destro è al massimo 1



NOTA perf. bil ⇒ bilanc.

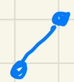
[1] Adelson-Velsky and Landis, 1962

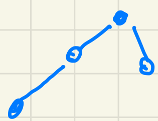
Alberi bilanciati in altezza: $\underbrace{\text{n° nodi}}_{n}$ vs $\underbrace{\text{altezza}}_{h}$

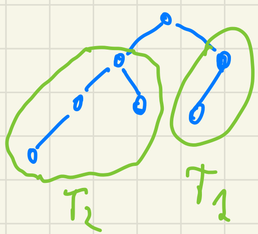
n° max di nodi → albero completo di altezza h: $n = 2^{h+1} - 1$

n° min di nodi

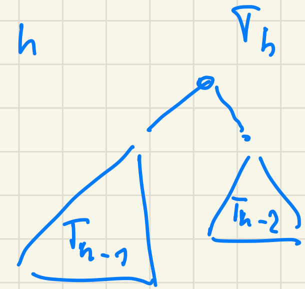
$h = 0$  T_0 $n_0 = 1$

$h = 1$  T_1 $n_1 = 2$

$h = 2$  T_2 $n_2 = 4$

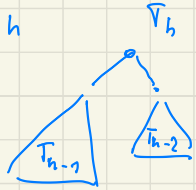
$h = 3$ 

T_3 $n_3 = 7$

h 

ALBERO di FIBONACCI
di ALTEZZA h

Albero AVL di
altezza minima



ALBERO di FIBONACCI
di ALTEZZA h

$n_h = \#$ nodi di T_h

$$n_h = \begin{cases} 1 & \text{se } h = 0 \\ 2 & \text{se } h = 1 \\ 1 + n_{h-1} + n_{h-2} & \text{altrim.} \end{cases}$$

Dim che

$$n_h = F_{h+3} - 1$$

$F_1 = F_2 = 1$

$F_k = F_{k-1} + F_{k-2}$

Ind.

$h = 1 \quad n_0 = 1 \quad F_3 = 2$

$h = 2 \quad n_1 = 2 \quad F_4 = 3$ | ok base

$\boxed{< h \to h}$

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

ip. ind

$$= \cancel{1} + \underbrace{F_{h+2} \cancel{-1} + F_{h+1}} - 1$$

$$= F_{h+3} - 1 \quad \square$$

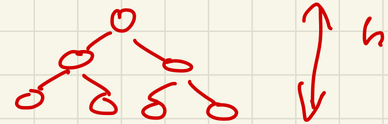
$$n_h \approx \frac{\phi^{h+3}}{\sqrt{5}} \qquad \sqrt{5}\, n_h \approx \phi^{h+3}$$

$$\log_\phi(\sqrt{5}\, n_h) \simeq h + 3 \qquad h \approx \log_\phi \sqrt{5} - 3 + \log_\phi(n_h)$$

$h = \Theta(\log n) \to$ Ricerca $\Theta(\log n)$

Alberi bilanciati in altezza: $\underbrace{\text{n° nodi}}_{n}$ vs $\underbrace{\text{altezza}}_{h}$

$h \rightarrow$ n° max nodi



albero completo

$n = 2^{h+1} - 1$

$h \rightarrow$ n° minimo di nodi : alberi di Fibonacci

$$n_h \approx \frac{\phi^{h+3}}{\sqrt{5}}$$

nodi albero di FB di altezza h

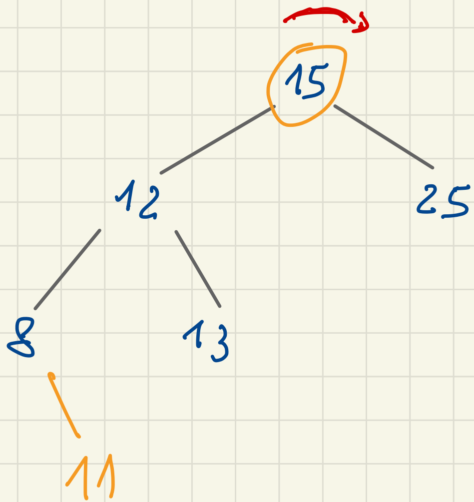
$$h = O(\log n)$$

$$h < 1.44 \log_2 n$$

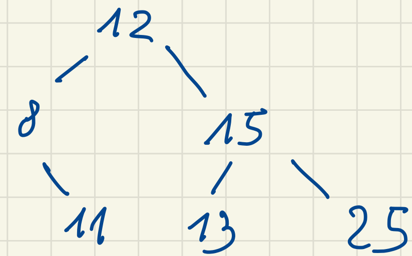
Alberi AVL: altezza $\Theta(\log n)$

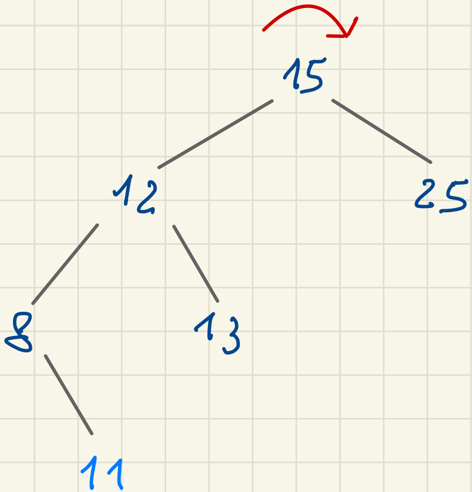
$\Rightarrow$ Ricerca costo $\Theta(\log n)$

Inserimento in alberi AVL

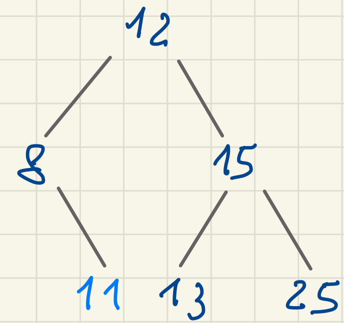


inserire 11

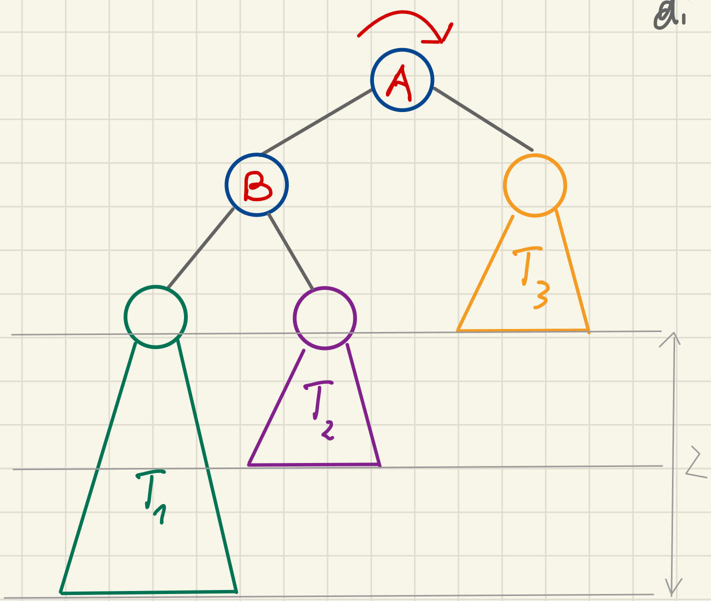




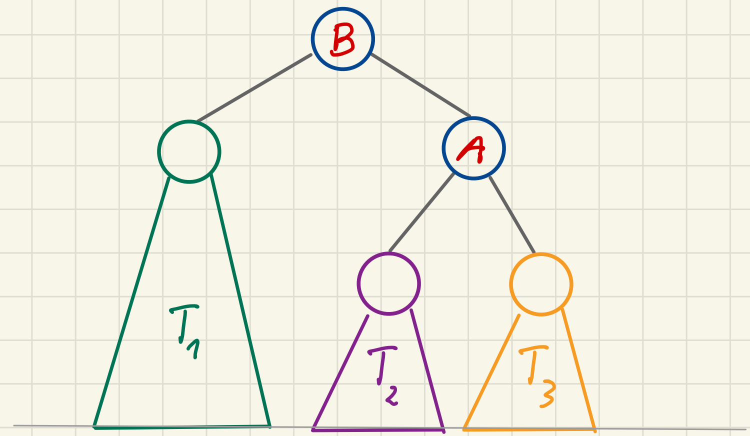
ROTAZIONE VS DESTRA



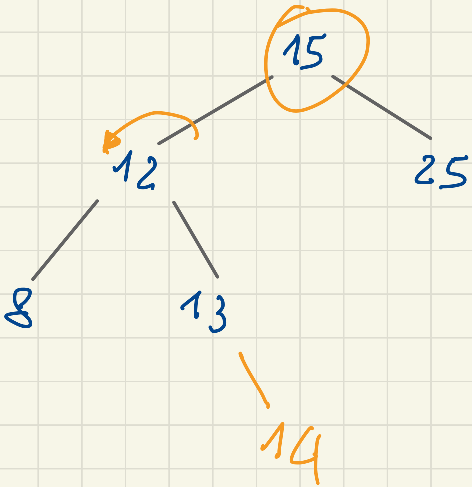
SBILANCIAMENTO a SX nel sottoalbero SX



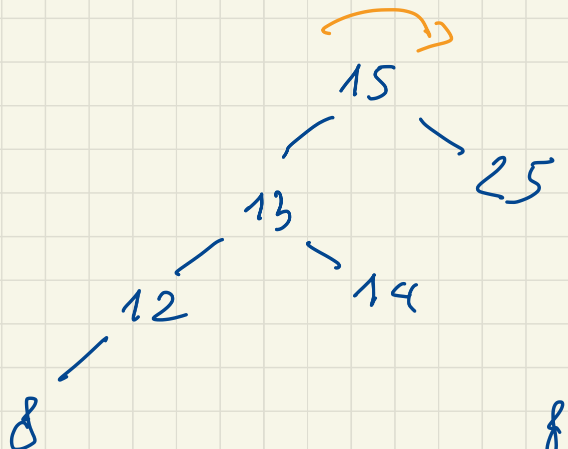
ROTAZIONE a DX
di perno A

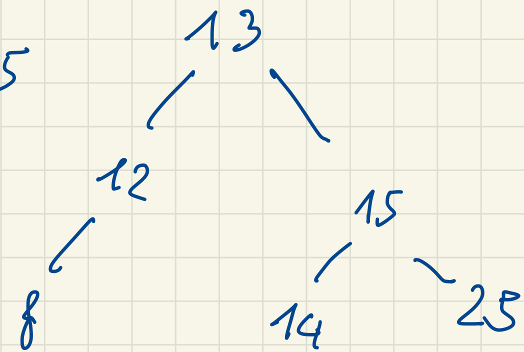


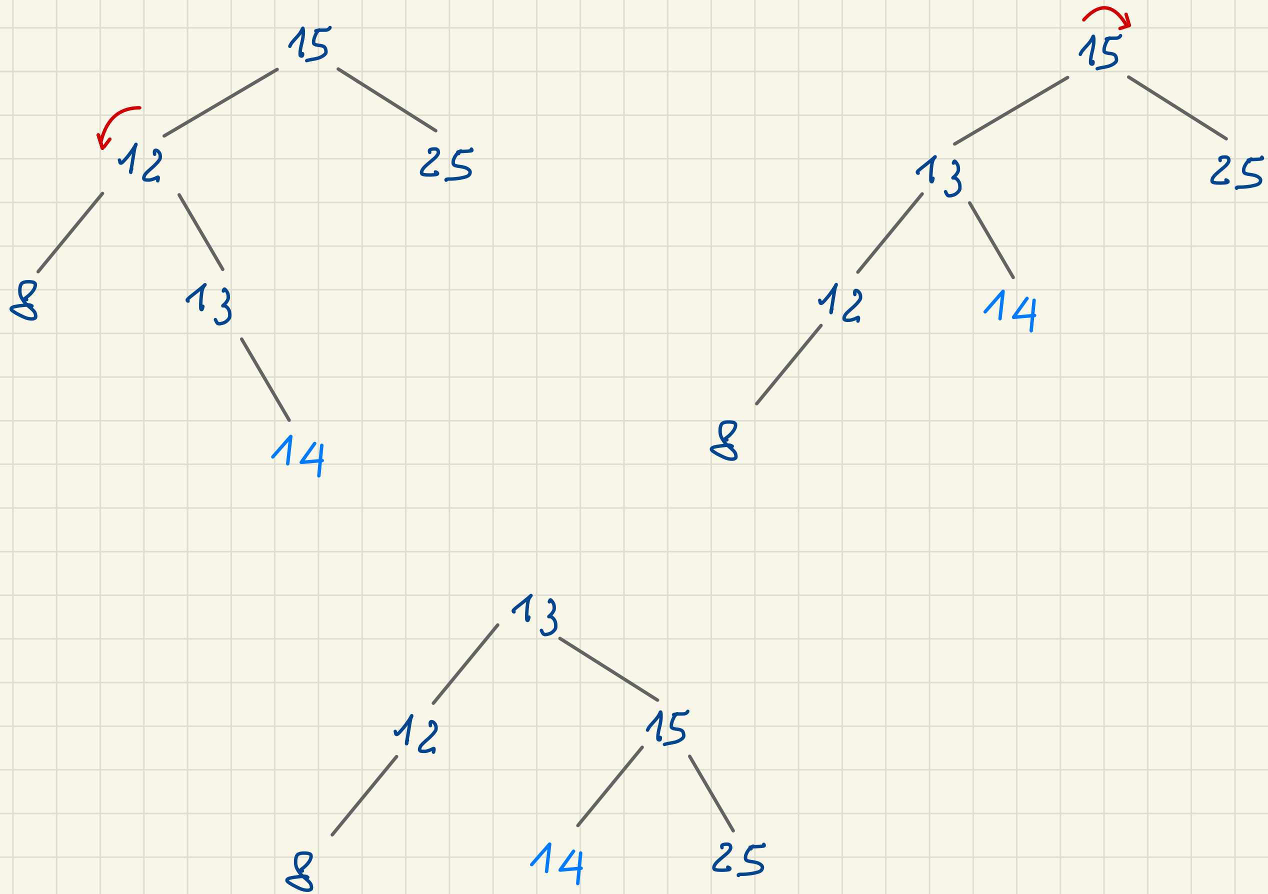
- Tempo rotazione $O(1)$
- Costo rot. inserimento $O(\log n)$



inserire 14



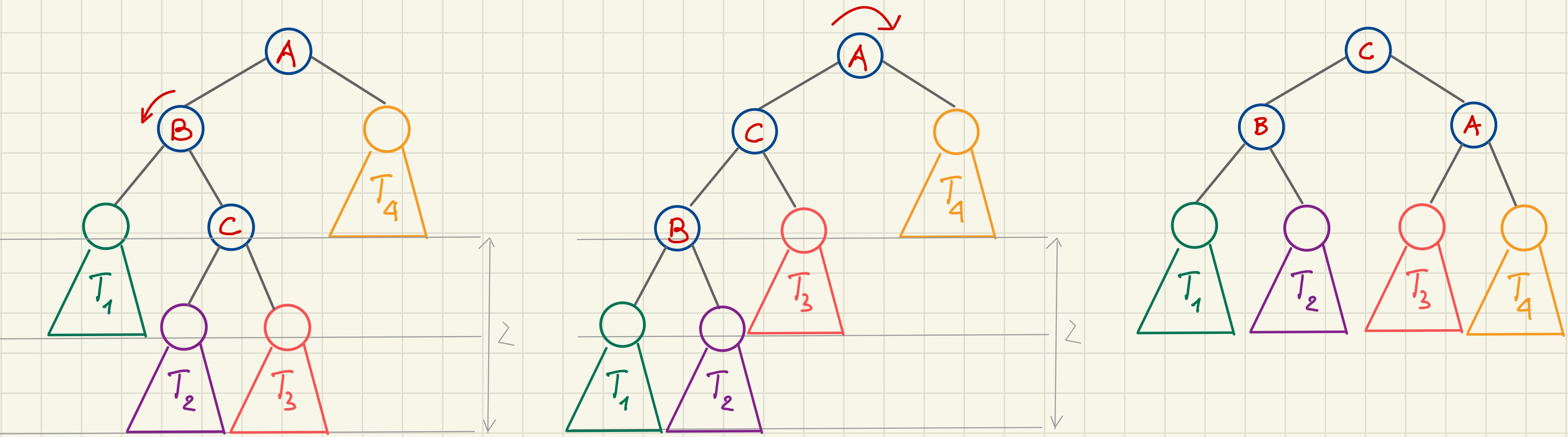



15
12
25
8
13
14
15
13
25
12
14
8
13
12
15
8
14
25

SBILANCIAMENTO a DX nel sottoalbero SX

1. ROTAZIONE a SX di perno B

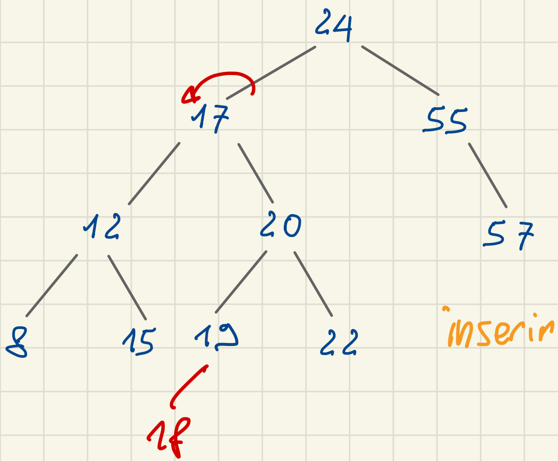
2. ROTAZIONE a DX di perno A



rotazioni tempo $O(1)$

Tempo inserimento $O(\log n)$

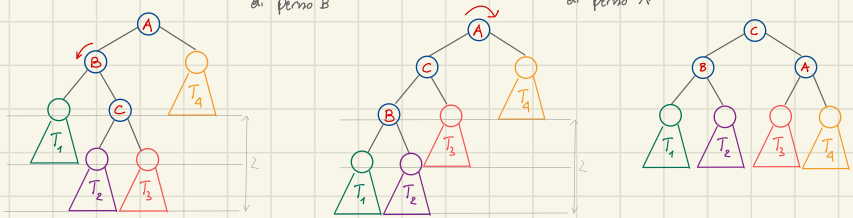
SBILANCIAMENTO a DX nel sottoalbero SX

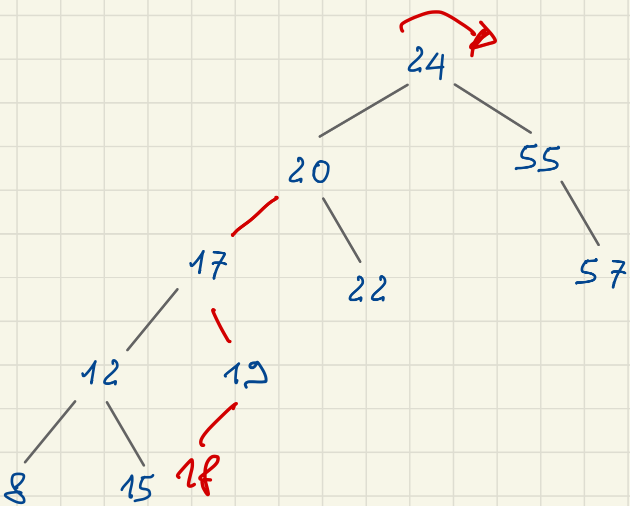


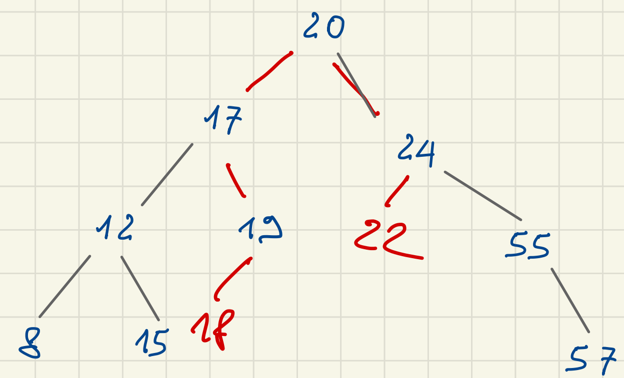
inserire 18

1. ROTAZIONE a SX di perno B

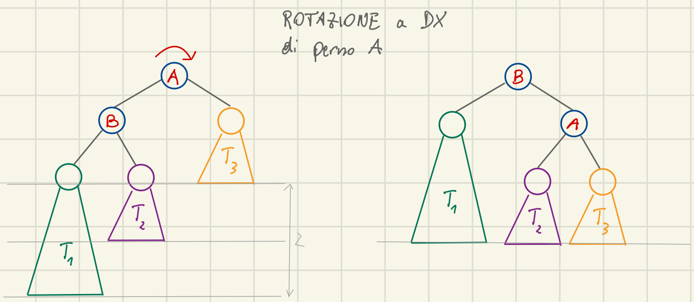
2. ROTAZIONE a DX di perno A



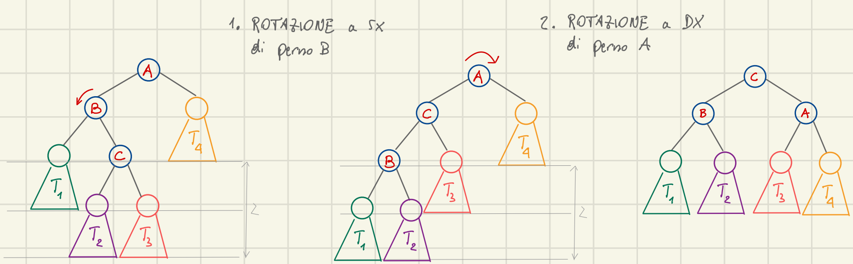




SBILANCIAMENTO a SX nel sottoalbero SX



SBILANCIAMENTO a DX nel sottoalbero SX



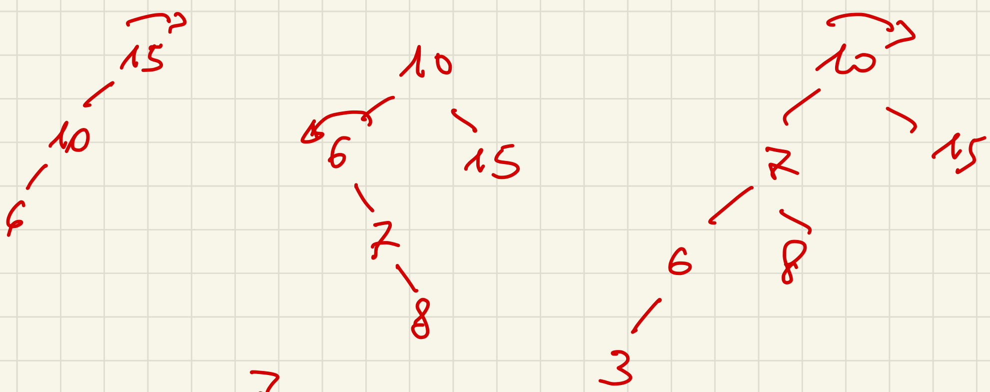
- Schemi simmetrici per sbilanciamenti nel sottoalbero dx

- Il ribilanciamento può essere effettuato in tempo costante

Esempio

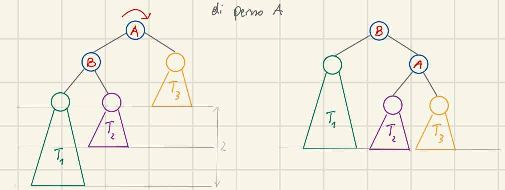
Disegnare l’albero AVL che si ottiene a partire da un albero vuoto inserendo uno dopo l’altro, nell’ordine indicato, i seguenti numeri:

15 10 6 7 8 3 4



SBILANCIAMENTO a SX nel sottoalbero SX

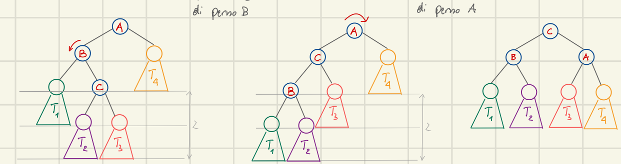
ROTAZIONE a DX
di perno A

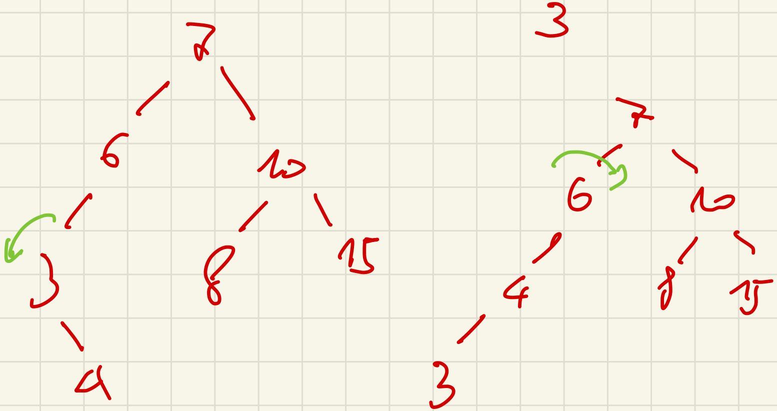


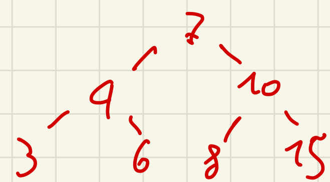
SBILANCIAMENTO a DX nel sottoalbero SX

1. ROTAZIONE a SX di perno B

2. ROTAZIONE a DX di perno A







ALBERI AVL

altezza $\Theta(\lg n)$

Tempo:

Ricerca $\Theta(\lg n)$

Inserimento $\Theta(\lg n)$

Cancellazione $\Theta(\lg n)$

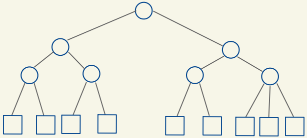
Alberi 2-3

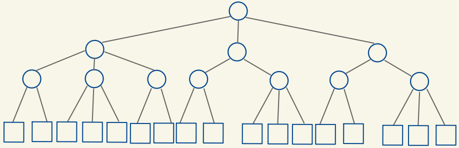
Alberi 2-3

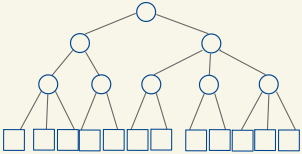
Definizione

Un *albero 2-3* è un albero in cui:

- ogni nodo interno ha 2 o 3 figli,
- tutte le foglie si trovano allo stesso livello.

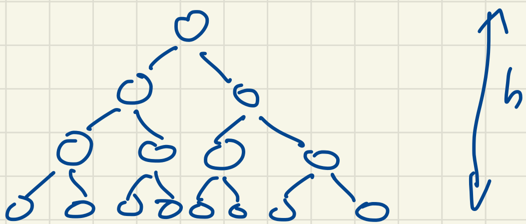






Alberi 2-3 : n° nodi/foglie vs $\underbrace{\text{altezza}}_{h}$

minimo n° nodi/foglie

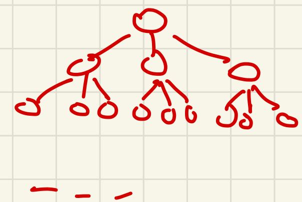


albero binario completo di altezza h

#nodi : $2^{h+1} - 1$

foglie : 2^h

max n° di nodi/foglie



nodi $\frac{3^{h+1}-1}{2}$

foglie 3^h

Alberi 2-3

Albero di altezza h

	minimo	massimo
numero nodi	$2^{h+1}-1$	$\frac{3^{h+1}-1}{2}$
numero foglie	2^h	3^h

$2^h \leq n \leq 3^h$

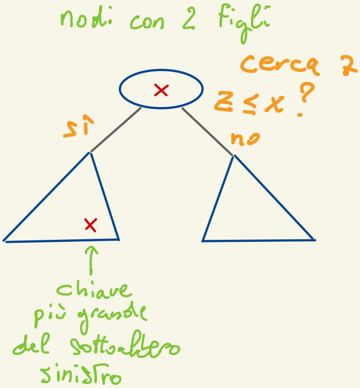
Dunque, se ci sono n foglie:

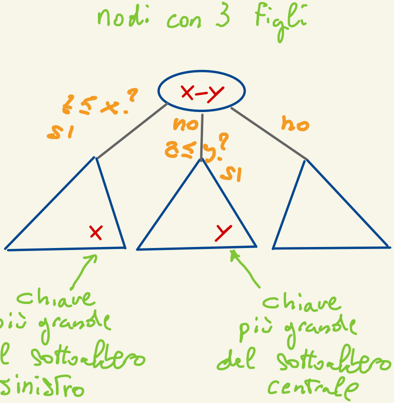
$$\log_3 n \leq h \leq \log_2 n$$

Pertanto, l'altezza è logaritmica rispetto al numero di foglie, $h = \Theta(\log n)$

Alberi 2-3 di ricerca

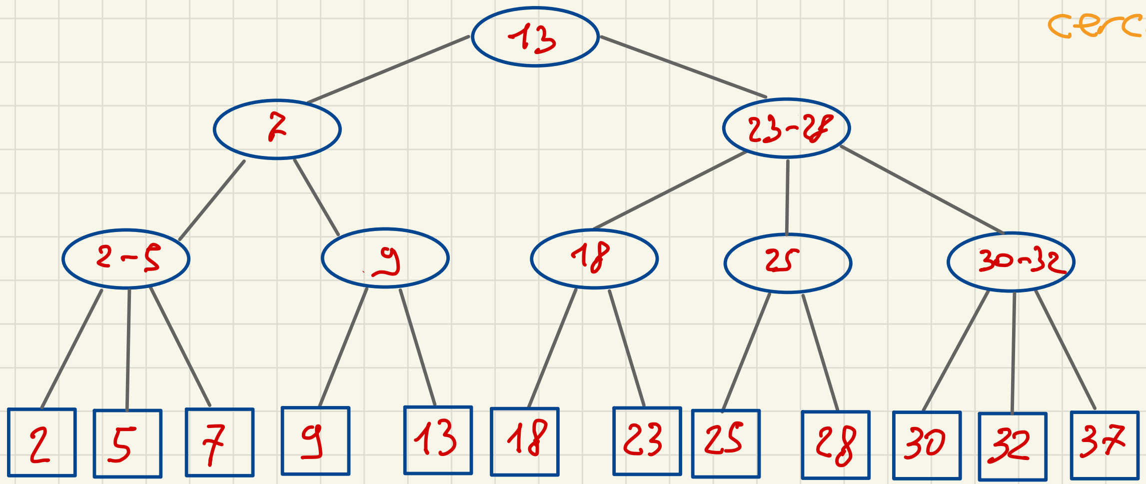
- Dati memorizzati *esclusivamente* nelle foglie, in ordine non decrescente da sinistra verso destra.
- I nodi interni contengono alcune chiavi, utilizzate come informazioni di *instradamento*.





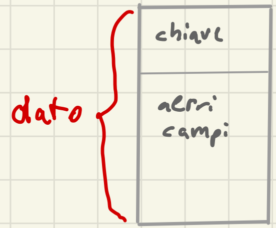
cerca 9

cerca 29

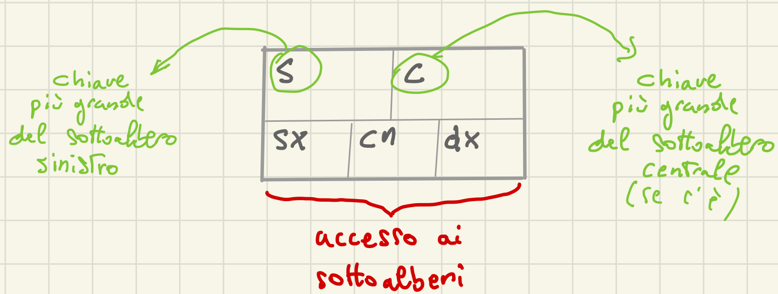


Rappresentazione

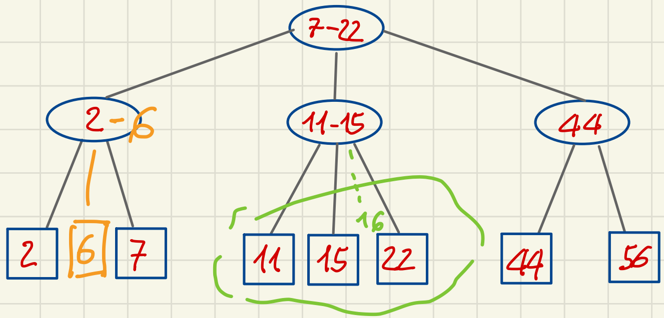
- FOGLIE



- NODI INTERNI

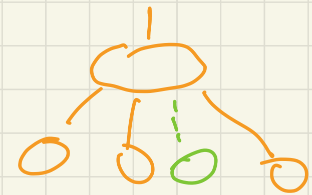


- Per inserimenti/cancellazioni è utile memorizzare in ogni nodo anche un puntatore al padre

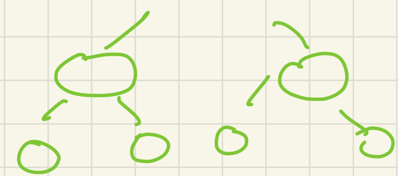


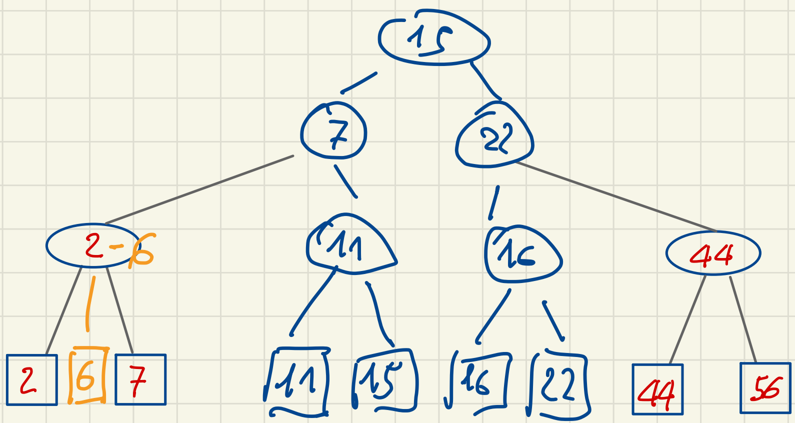
inserisci 6

inserisci 16



split

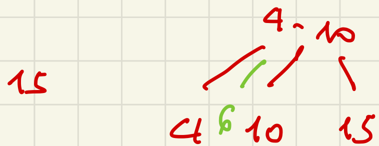


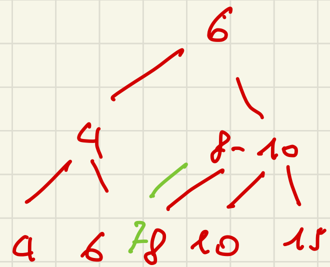


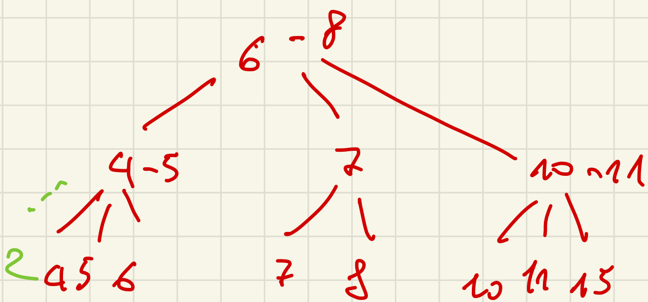
inserimento $\Theta(\log n)$ passi

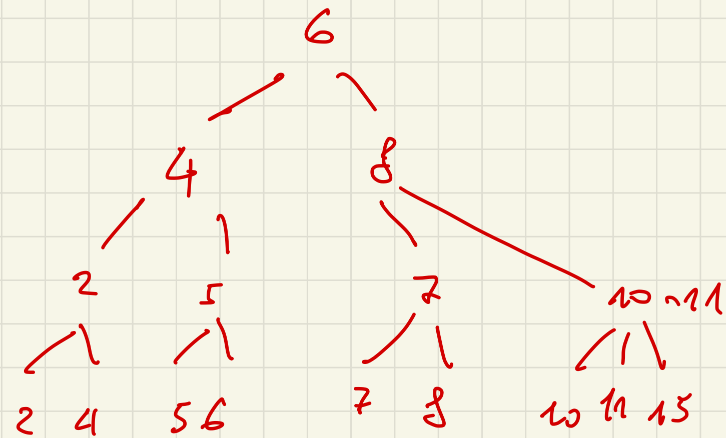
Esempio

15 10 4 6 8 7 5 11 2









Alberi 2-3: costo operazioni

- Ricerca $\Theta(\log n)$
- Inserimento $\Theta(\log n)$
- Cancellazione $\Theta(\log n)$