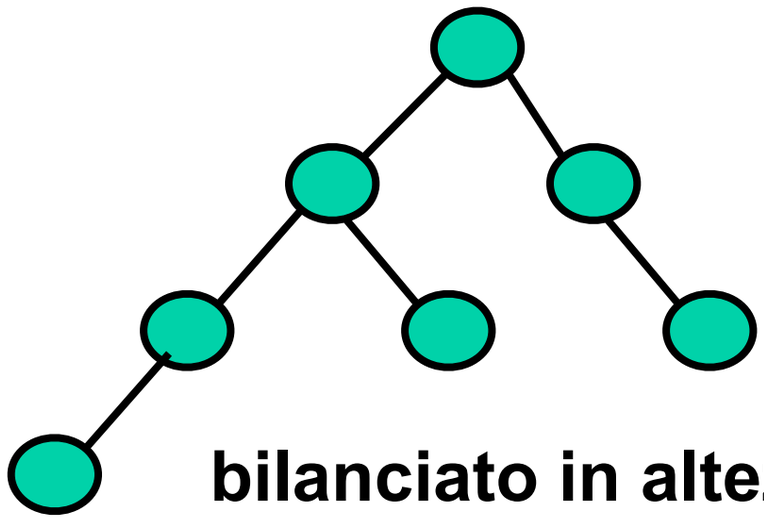
ABR di altezza logaritmica

Esistono vari criteri di bilanciamento di alberi binari di ricerca che ne garantiscono l'altezza logaritmica.

Chiamiamo bilanciato ogni albero binario di altezza logaritmica nel numero dei nodi.

ABR bilanciati in altezza

Un ABR si dice bilanciato in altezza se per ogni nodo v la differenza in valore assoluto tra l'altezza del sottoalbero sinistro di v e l' altezza del sottoalbero destro di v è ≤ 1.



bilanciato in altezza

Gli ABR bilanciati in altezza si chiamano brevemente alberi AVL, acronimo dagli autori Georgy Maximovich Adel'son-Vel'skii e Yevgeniy Mikhailovich Landis che li hanno introdotti nel 1962.

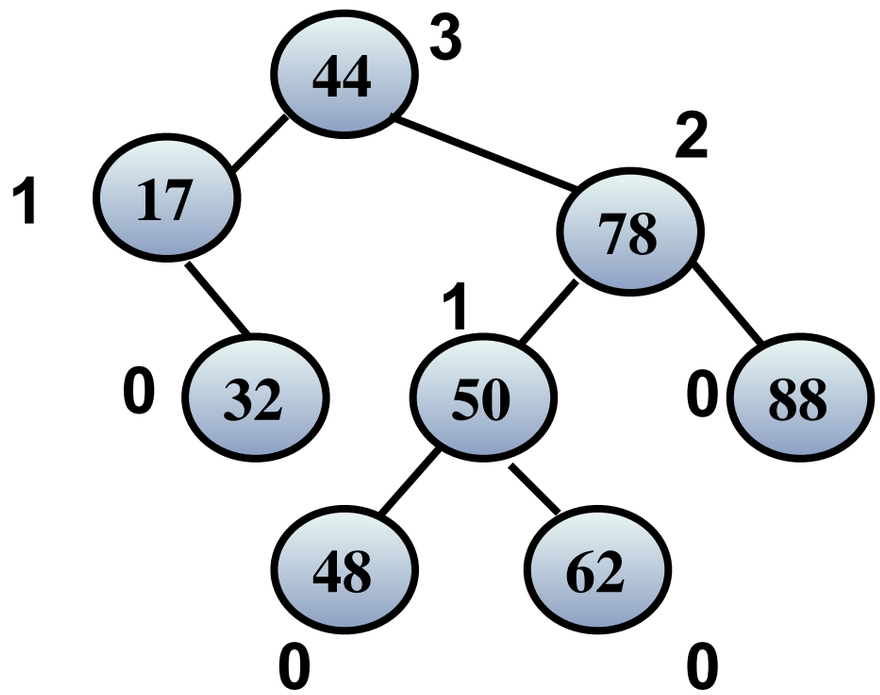
N.B. Prendiamo come altezza il massimo numero di archi in un cammino radice-foglia.

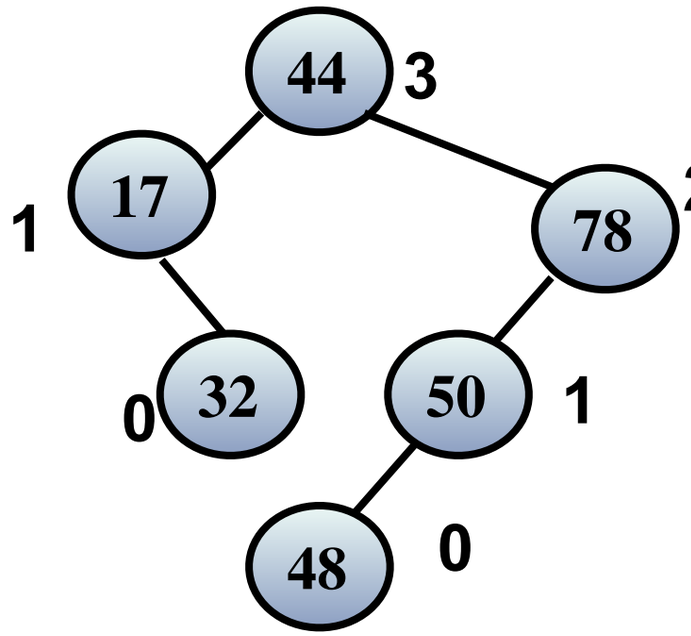
Quindi l'albero vuoto ha altezza -1.

Alberi AVL: esempi

bilanciamento in altezza:

per ogni nodo la differenza in valore assoluto tra l'altezza del sottoalbero sinistro di v e quella del sottoalbero destro di v è ≤ 1



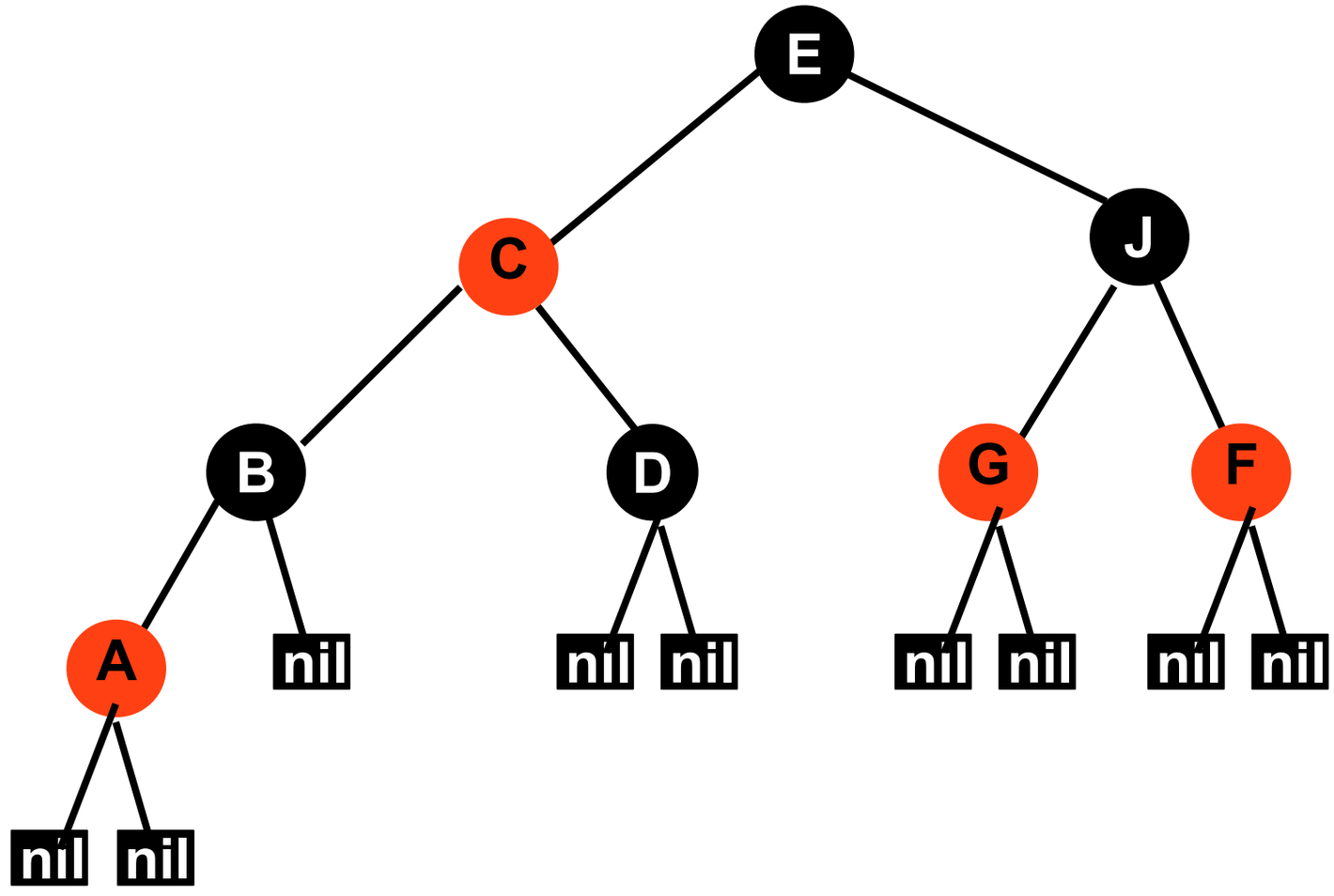


NON bilanciato: il nodo 78 ha un sotto albero destro di altezza -1, mentre il sotto albero sinistro ha altezza 1, quindi la differenza è 2.

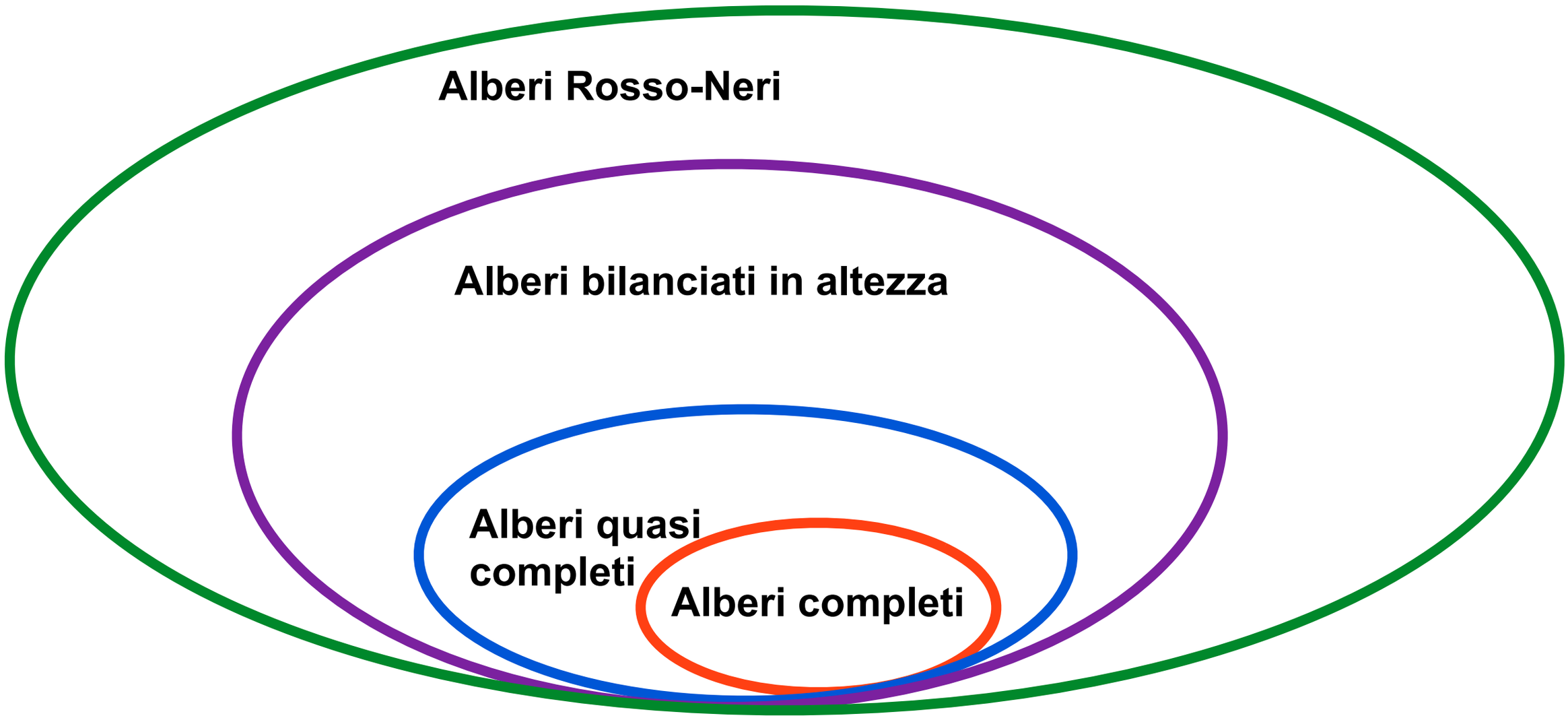
ABR Rosso Neri

- **alberi rosso-neri, introdotti nel 1972 da Rudolf Bayer, professore alla Technische Universität München, con il nome di "B-alberi simmetrici". Leonidas J. Guibas, Professore alla Stanford University e di Robert Sedgewick, professore alla Princeton University ne provano molte proprietà, chiamandoli alberi rosso-neri, in un successivo articolo, apparso nel 1978.**

Ogni nodo in un albero rosso-nero è colorato di rosso o di nero, ma la colorazione deve soddisfare dei vincoli che garantiscono che nessun percorso radice-foglia è lungo più del doppio di ogni altro, così l'albero è abbastanza bilanciato da avere altezza logaritmica nel numero dei nodi.



Relazioni tra classi di alberi



Le operazioni

Perchè le operazioni di ricerca, inserimento e cancellazione in un ABR siano di complessità O(log n), dove n è il numero dei nodi dell'albero bisogna modificarle in modo che le eventuali operazioni di ribilanciamento dell'albero siano di complessità O(log n).

Gli alberi bilanciati in altezza o alberi AVL (Adel'son-Vel'skii, Landis, 1962), sono i primi per i quali si ottiene questo risultato poi ottenuto anche per gli alberi rosso-neri (Bayer,1972) e dagli Splay trees (Sleator,Tarjan 1985), che hanno la proprietà che i nodi ricercati vengono spostati verso la radice quindi saranno ricercati più efficientemente in seguito.

Esistono vari altri alberi di ricerca (anche non binari) con i quali possiamo implementare un dizionario di n elementi con operazioi in tempo O(log n).

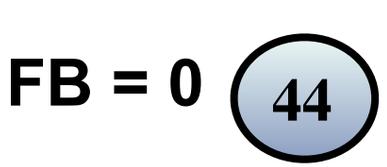
Alcuni di questi, in particolare

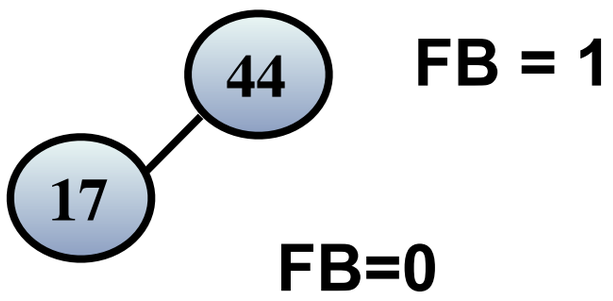
2-3 alberi
2-3-4-alberi
B-alberi

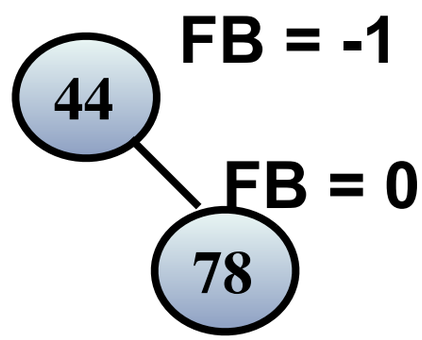
} **si trasformano in alberi rosso-neri**

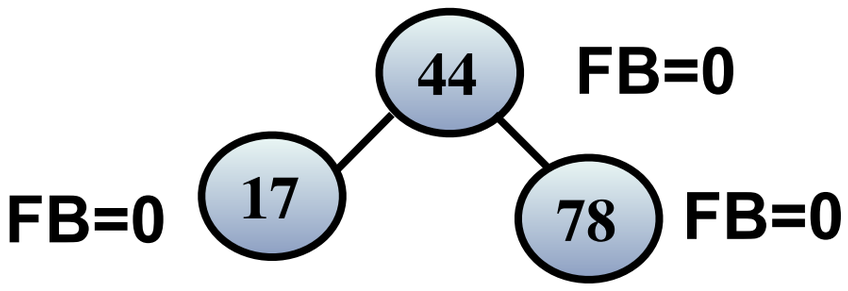
Costruzione AVL

Detto FB il fattore di bilanciamento di un nodo, cioè la differenza tra altezza sotto albero sinistro e quella del destro, vediamo alcuni esempi di AVL, cominciando dai più piccoli in termini di numero dei nodi:

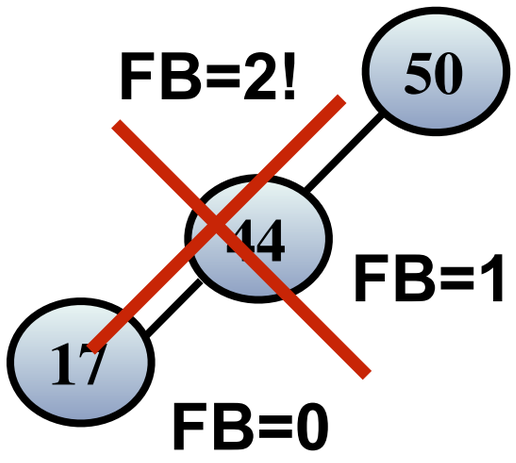




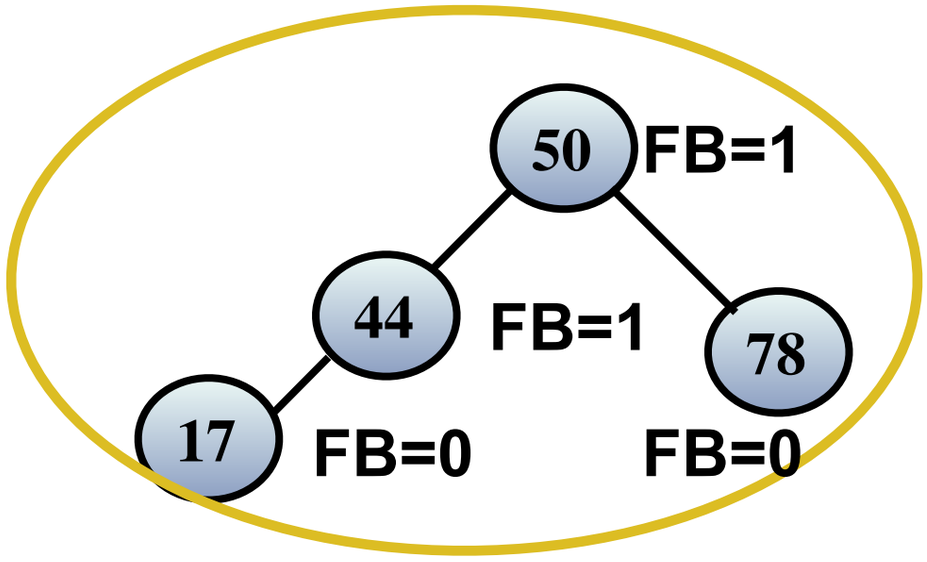


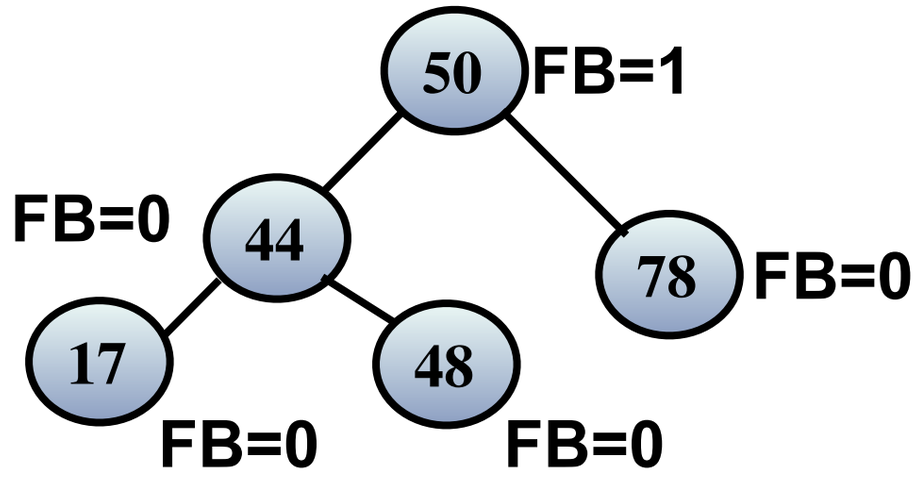


L’unico AVL con tre nodi!

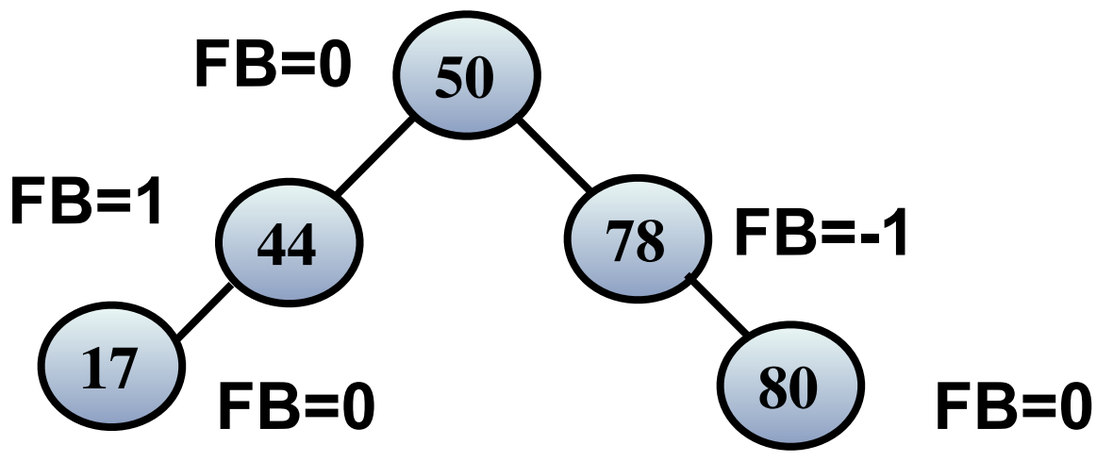


Esempi di AVL



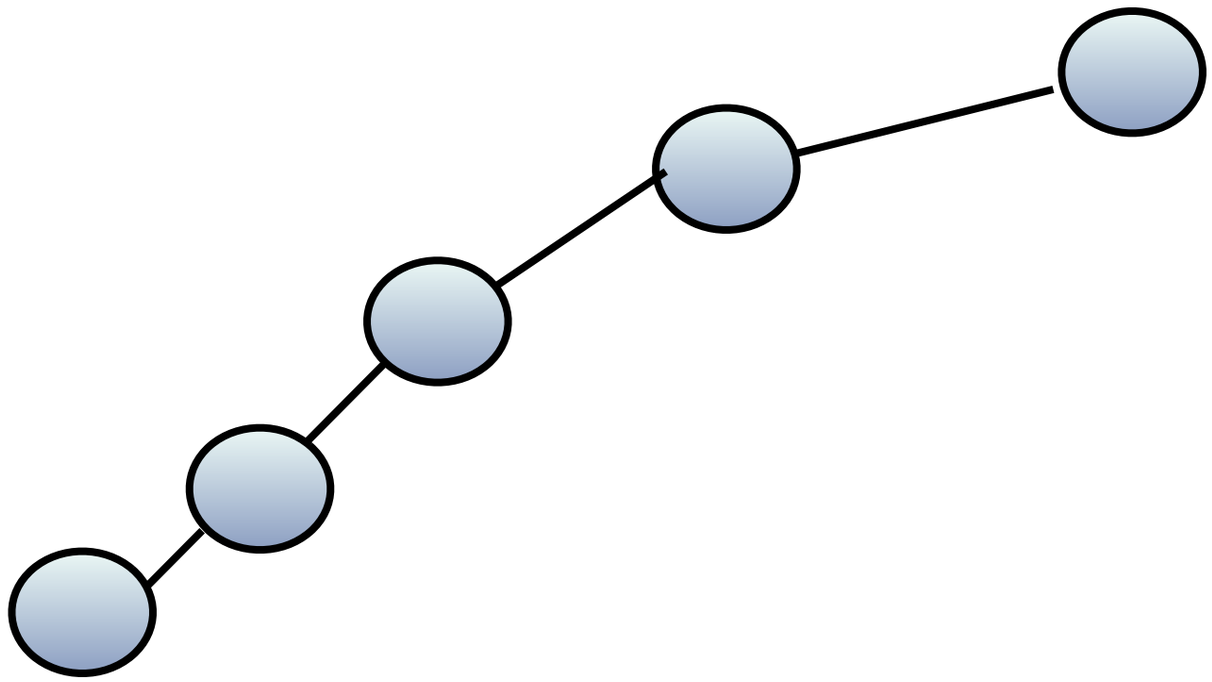


tra quelli di altezza 3 ha il minor numero di nodi



Massima altezza, minimo nodi

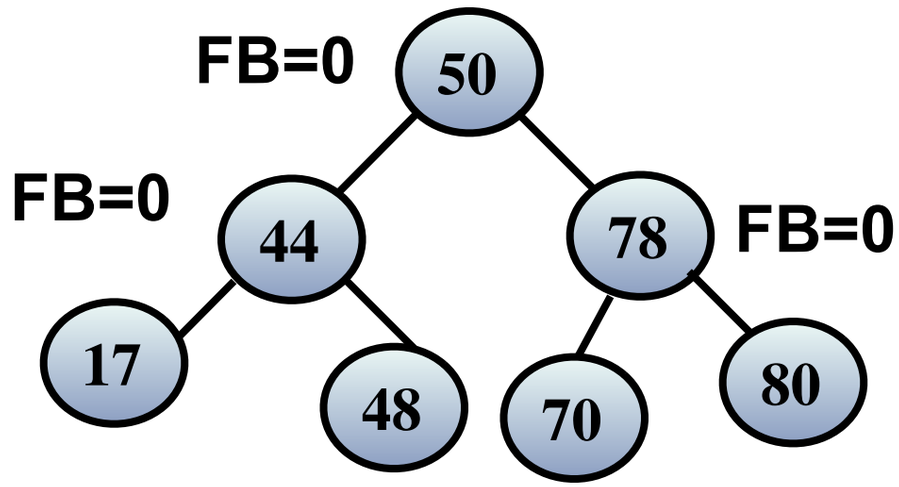
Ricordiamo che un albero degenere è il più sbilanciato tra tutti gli alberi binari ed è quello di altezza massima a parità di numero di nodi, o anche è quello che a parità di altezza ha il minimo numero di nodi (un solo nodo su ogni livello).



Massimo nodi, minima altezza

L’albero completo è un AVL di altezza logaritmica, e tutti i suoi nodi interni hanno FB = 0

(le foglie avendo sempre FB =0).



É l’albero AVL che a parità di altezza ha il massimo numero di nodi o anche quello che a parità di numero di nodi ha altezza minima.

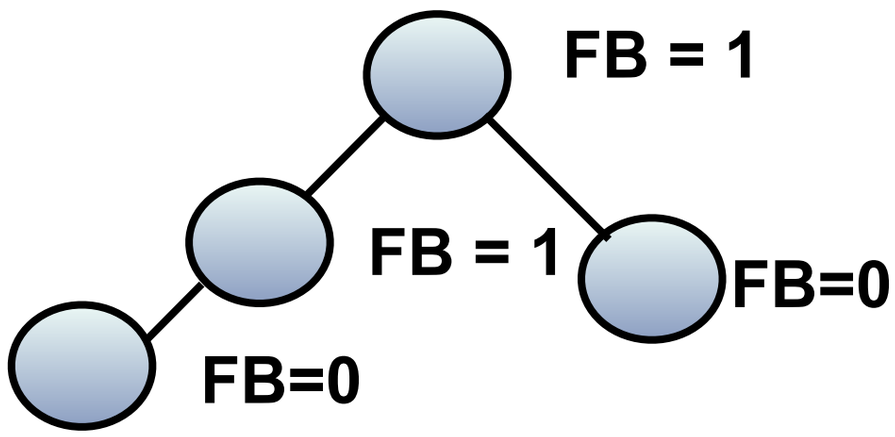
Massima altezza, minimo nodi per AVL

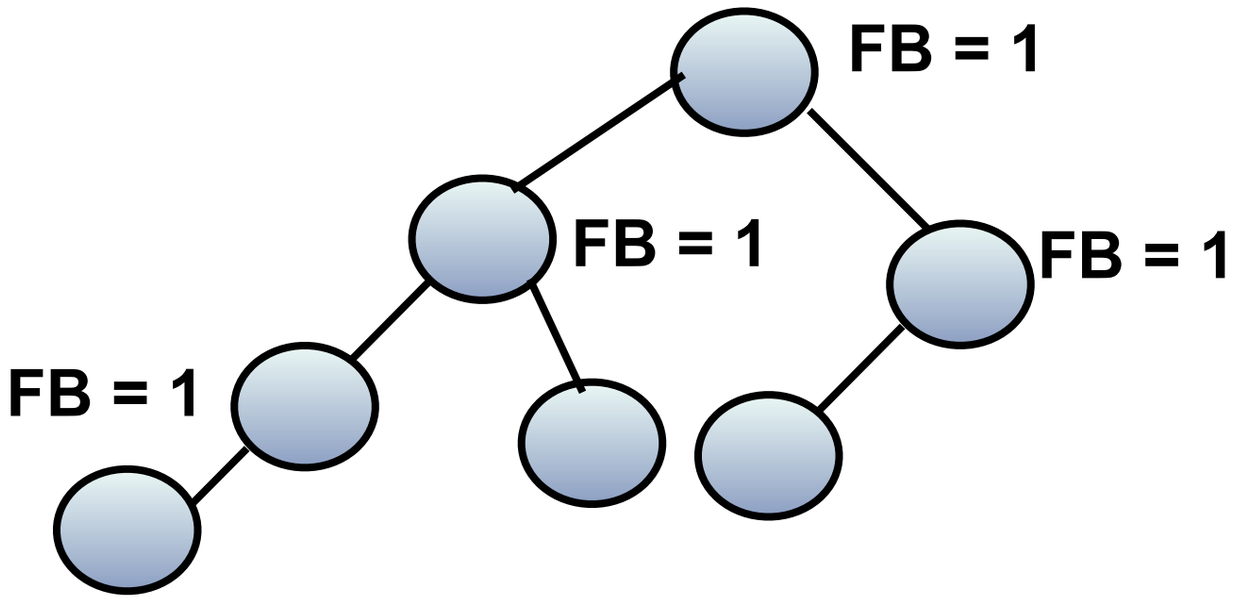
Cerchiamo di determinare gli AVL più sbilanciati che possiamo costruire, cioè cerchiamo gli AVL che a parità di altezza hanno il minimo numero di nodi.

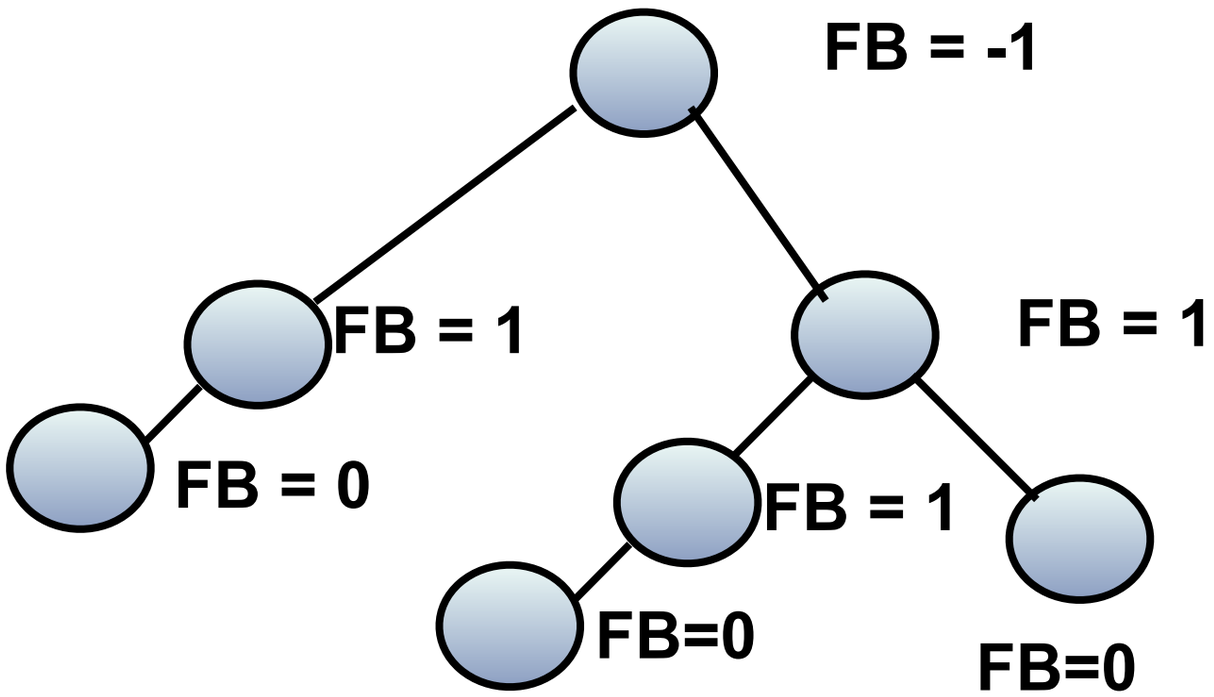
Il massimo sbilanciamento si ottiene imponendo a tutti i nodi non foglia di avere un FB diverso da 0.

Per semplicità stabiliamo che debba essere 1, se prendessimo sempre -1 si avrebbe un risultato equivalente.

Alberi AVL “sbilanciati”

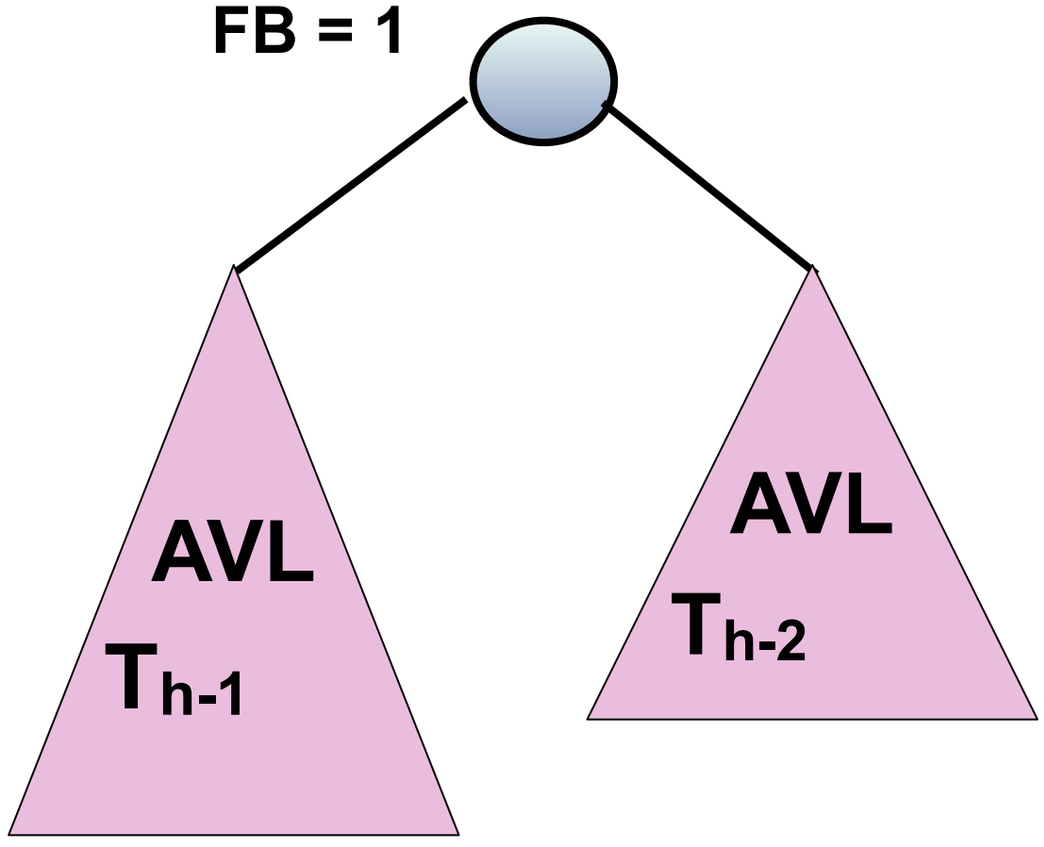






Una scelta “mista” è solo più difficile da gestire, volendo determinare il rapporto tra altezza e numero dei nodi.

Alberi AVL: costruzione dei più sbilanciati



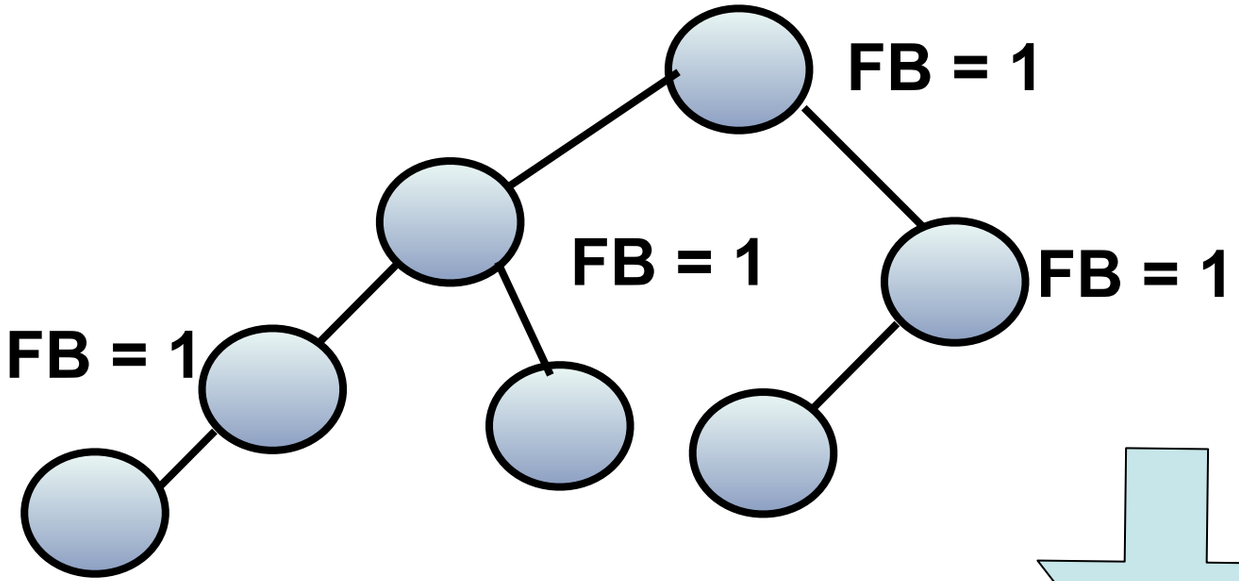
La regola di costruzione è:

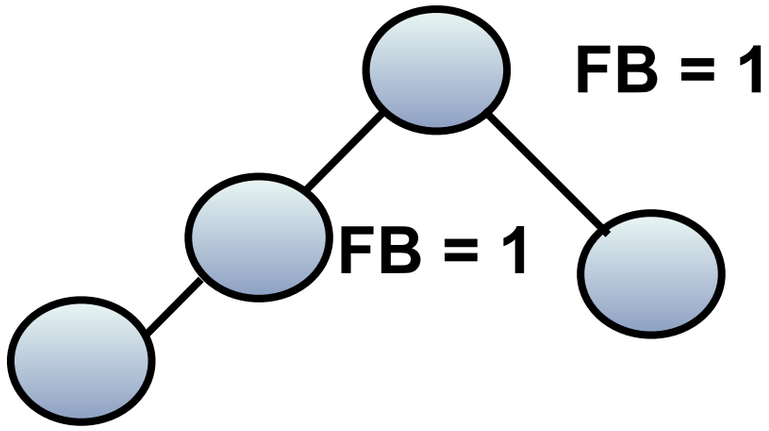
Base: l'albero vuoto è un AVL più sbilanciato a sinistra di altezza -1, T_{-1}, l'albero con un solo nodo è un AVL più sbilanciato a sinistra di altezza 0, T_0,

Passo induttivo: dati due AVL più sbilanciato a sinistra T_{h-1} e T_{h-2}, un nuovo AVL di altezza h si costruisce prendendo un nuovo nodo come radice e come sotto albero sinistro T_{h-1}, l'AVL più sbilanciato a sinistra di altezza h-1 e come sotto albero destro T_{h-2}, quello di altezza h-2.

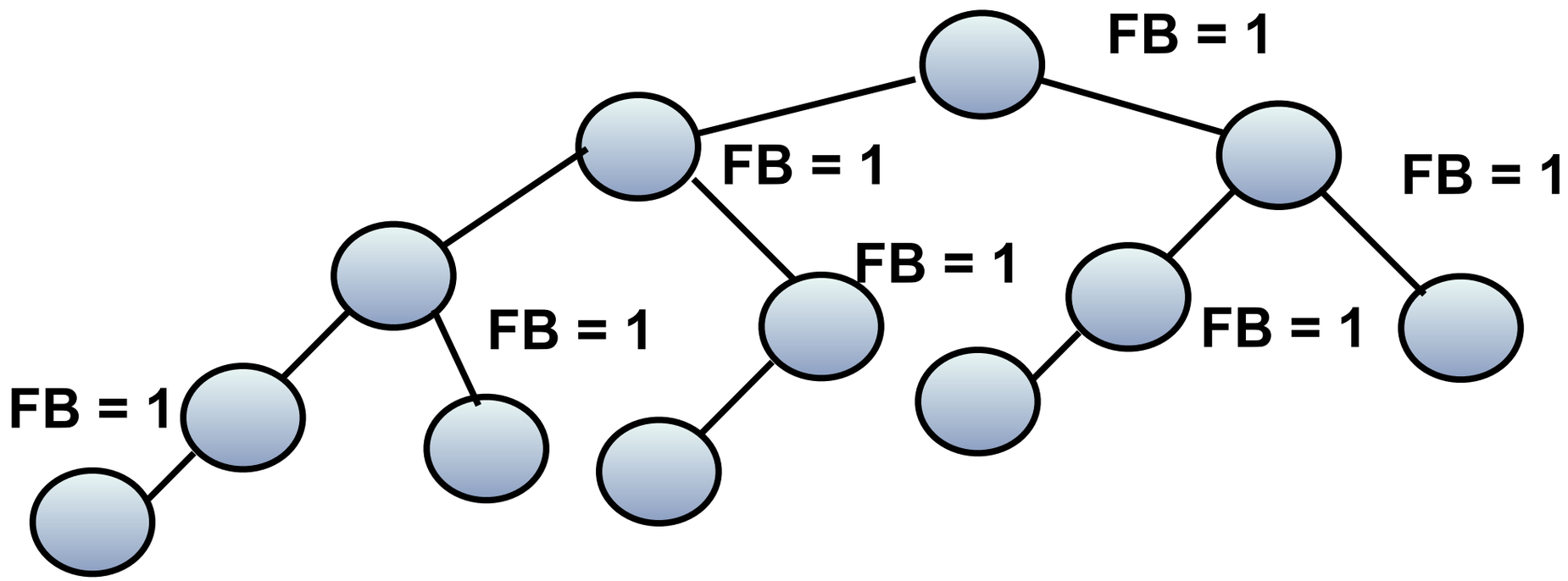
Per costruzione tutti i nodi interni di un AVL più sbilanciato a sinistra hanno FB = 1

Esempio costruzione dei più sbilanciati





ogni nodo interno ha un FB = + 1!

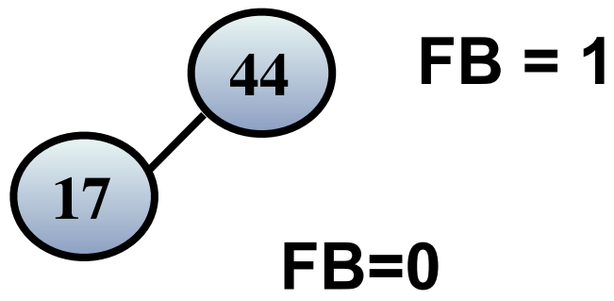


Alberi AVL: altezza

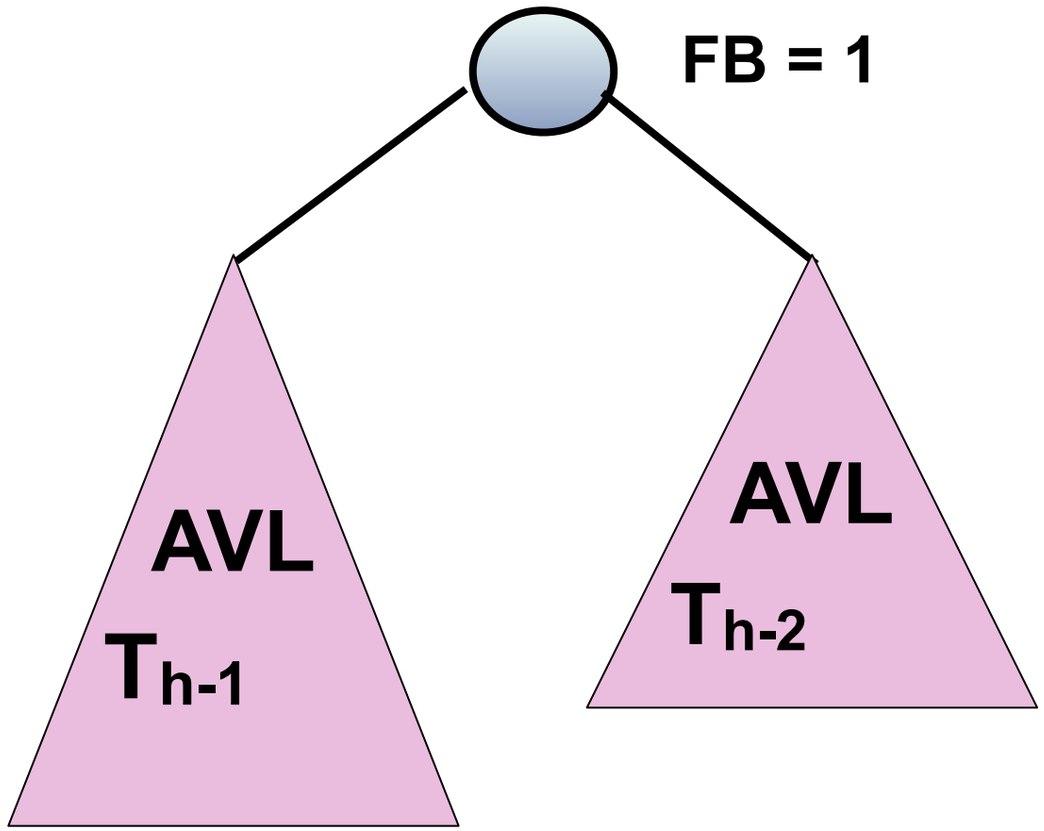
Il più piccolo albero AVL più sbilanciato a sinistra non vuoto è

l'albero T_0 con $h_1 = 0$ e $n_1 = 1$

l'albero T_1 con $h_1 = 0$ e $n_1 = 2$



Alberi più sbilanciati calcolo numero dei nodi



Numero nodi in un AVL più sbilanciato a sinistra di altezza h:

$n_0 = 1$

$n_1 = 2$

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

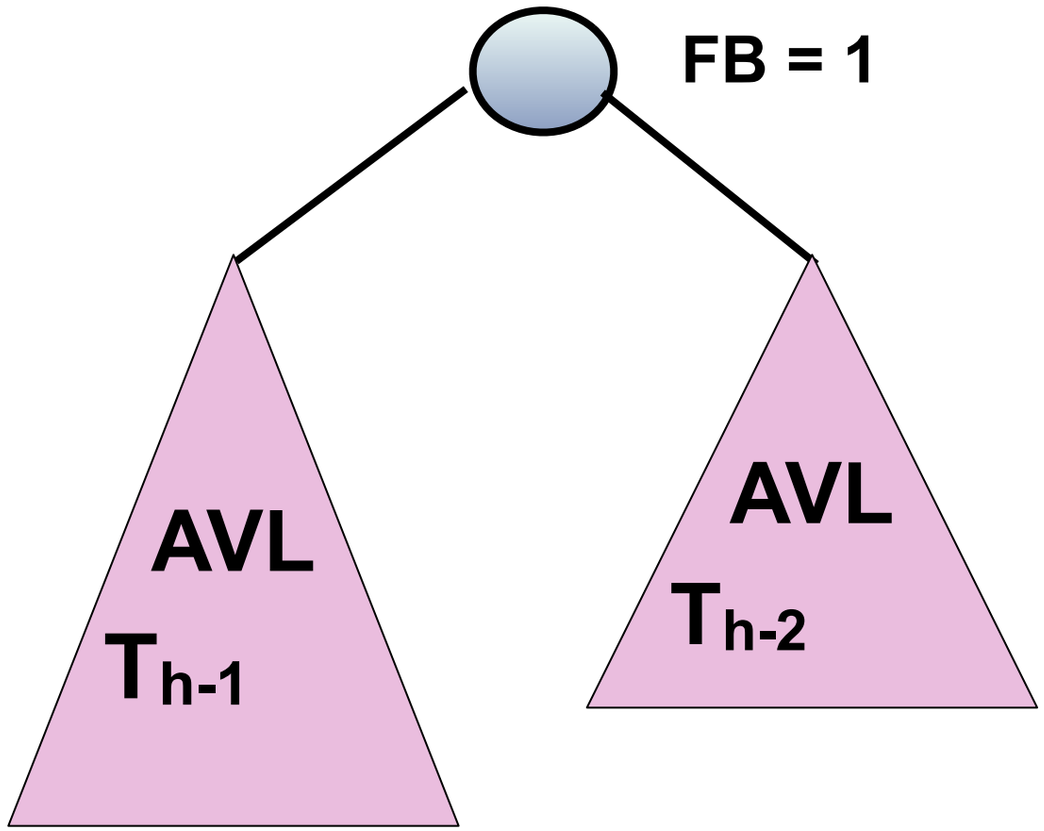
Fibonacci

$Fib_1 = 1$

$Fib_2 = 1$

$Fib_h = Fib_{h-1} + Fib_{h-2}$

Alberi di Fibonacci calcolo numero dei nodi



Numero nodi in un AVL più sbilanciato a sinistra di altezza h:

$n_0 = 1$

$n_1 = 2$

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

Fibonacci

$Fib_1 = 1$

$Fib_2 = 1$

$Fib_h = Fib_{h-1} + Fib_{h-2}$

$Fib_1 = 1$ e $n_1 = 2$

Per $n \geq 1$, si ha $Fib_h < n_h$

Alberi di Fibonacci: conclusione

Sappiamo che $Fib_h > \varphi^h/5^{1/2}$ dove $\varphi = 1{,}618\ldots$ è la sezione aurea e

usando la relazione $Fib_h < n_h$, per $h \geq 1$, otteniamo che

$$\varphi^h/5^{1/2} < Fib_h < n_h$$

Se prendiamo un albero AVL qualunque di altezza h e con n nodi allora $n > n_h$ e

$$\varphi^h/5^{1/2} < n$$

quindi applicando il logaritmo

$$h - \log_\varphi 5^{1/2} < \log_\varphi n \quad \text{e quindi}$$

$$h < \log_\varphi n + \log_\varphi 5^{1/2} = O(\lg n)$$

Alberi AVL: altezza in O(lgn)

Per gli alberi AVL abbiamo dimostrato

1. che la loro altezza è in O(lg n),

2. che gli alberi di Fibonacci che sono una classe di alberi AVL che a parità di altezza hanno il minimo numero di nodi.

Poiché per un qualsiasi albero binario T di altezza h si ha che h = Ω(lg n),

concludiamo che se T è un qualsiasi albero AVL di altezza h, vale h = θ(lg n)

Alberi AVL: altezza in O(lgn)

Senza ricorrere alla crescita dei numeri di Fibonacci possiamo ottenere il risultato cercato esaminando direttamente la ricorrenza

$$n_h = n_{h-1} + n_{h-2} + 1 \text{ per } h>2$$

con casi base

$$n_1 = 1 \text{ e } n_2 = 2$$

Poiché $n_{h-1} > n_{h-2} + 1$

($n_3 = n_1 + n_2 + 1 = 4$ e l'albero di altezza 3 è l'ultimo che si può costruire aggiungendo solo un nodo nel sottoalbero sinistro)

$$n_h > 2n_{h-2} \text{ per } h>2$$

quindi

$$n_h > 2n_{h-2} > 2^2 n_{h-4} > \ldots > 2^i n_{h-2i}$$

Da cui $n_h > 2^{h/2}$. Come prima osservando che se T è un AVL qualsiasi con n nodi e di altezza h si ha $n > n_h$, passando al logaritmo si conclude che lg (n) > h/2 e quindi che h = O(lg n).

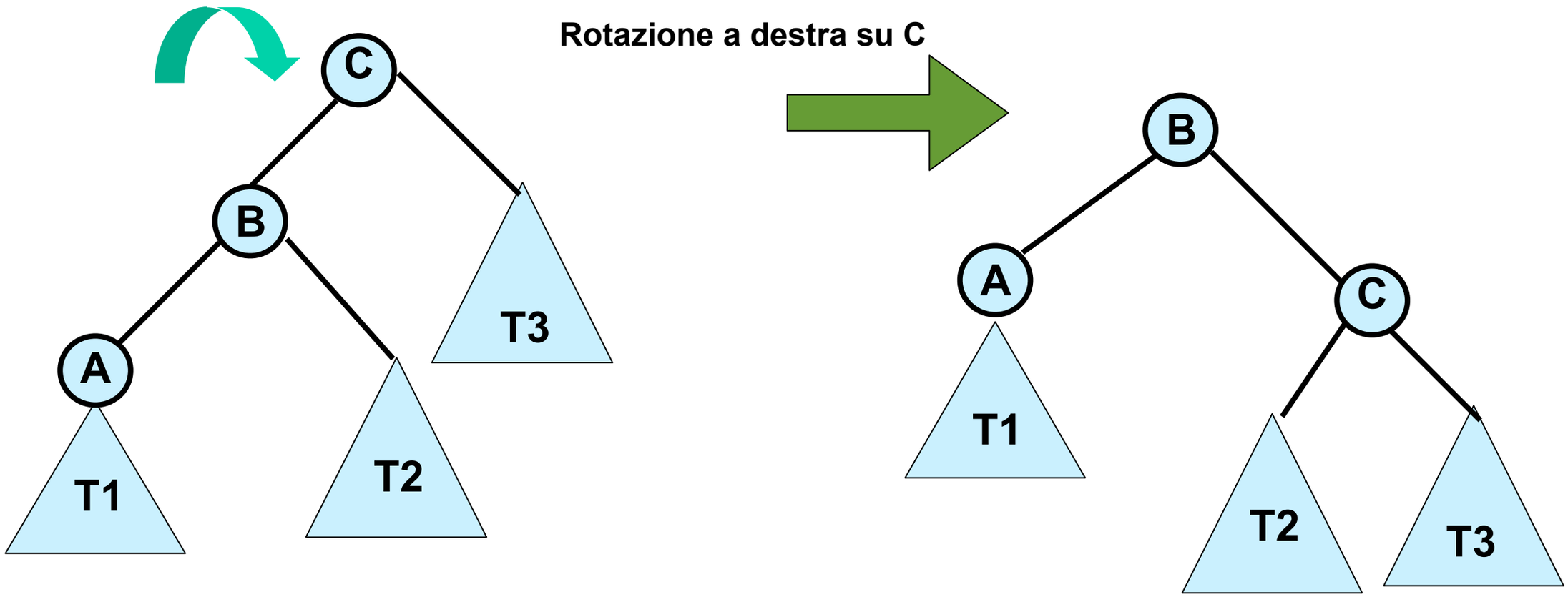
Operazioni di ribilanciamento

Le operazioni di inserimento e cancellazione sono le stesse che negli ABR qualunque, ma modificate in modo da poter eventualmente ribilanciare l'albero.

Descriviamo nel seguito le operazioni sugli alberi binari che ci consentiranno di ripristinare il bilanciamento in altezza, eventualmente perso in seguito a cancellazioni o inserimenti.

Si tratta di rotazioni.

Rotazioni su ABR

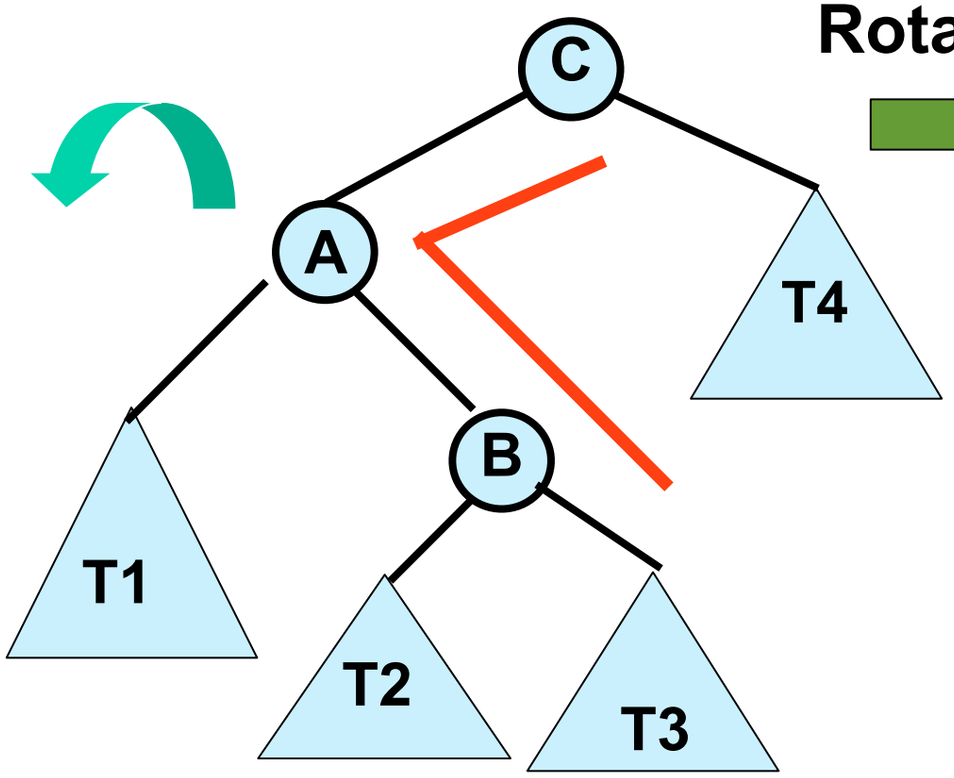


Dopo la rotazione l'albero è ancora un ABR, perchè se, b è un nodo in T2 e c in T3 allora vale

A < B < b < C < c .

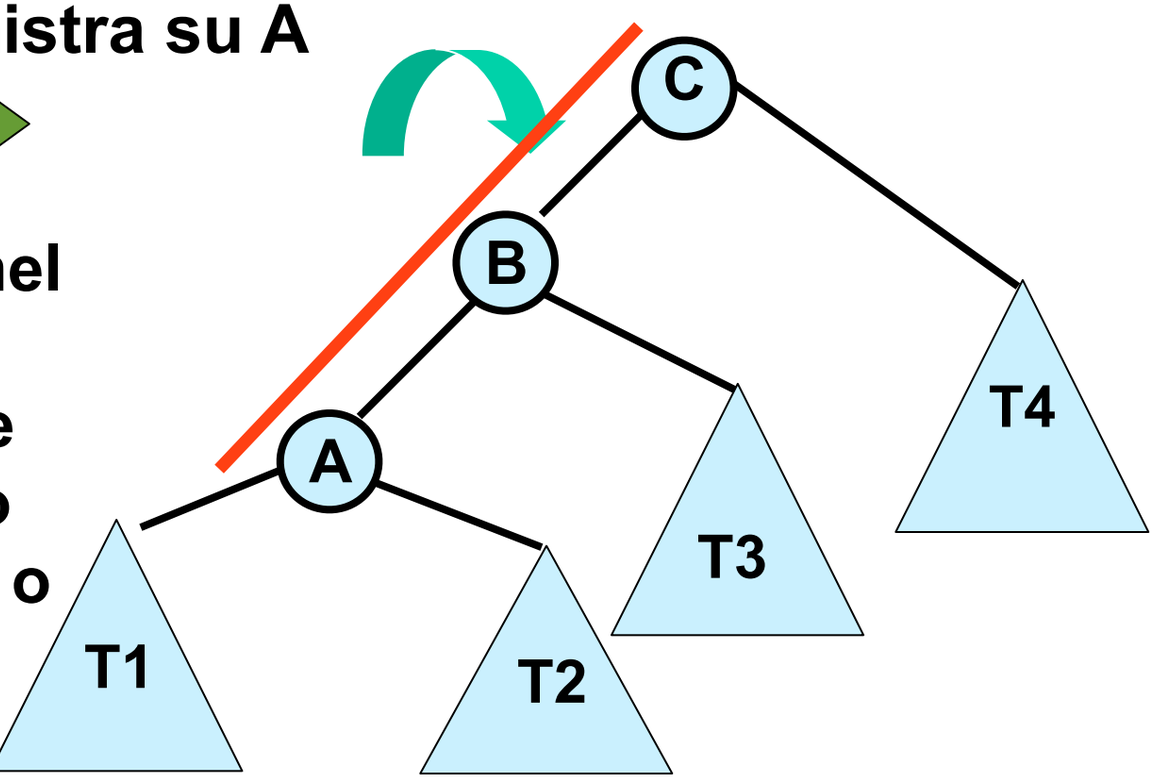
Richiede tempo O(1)

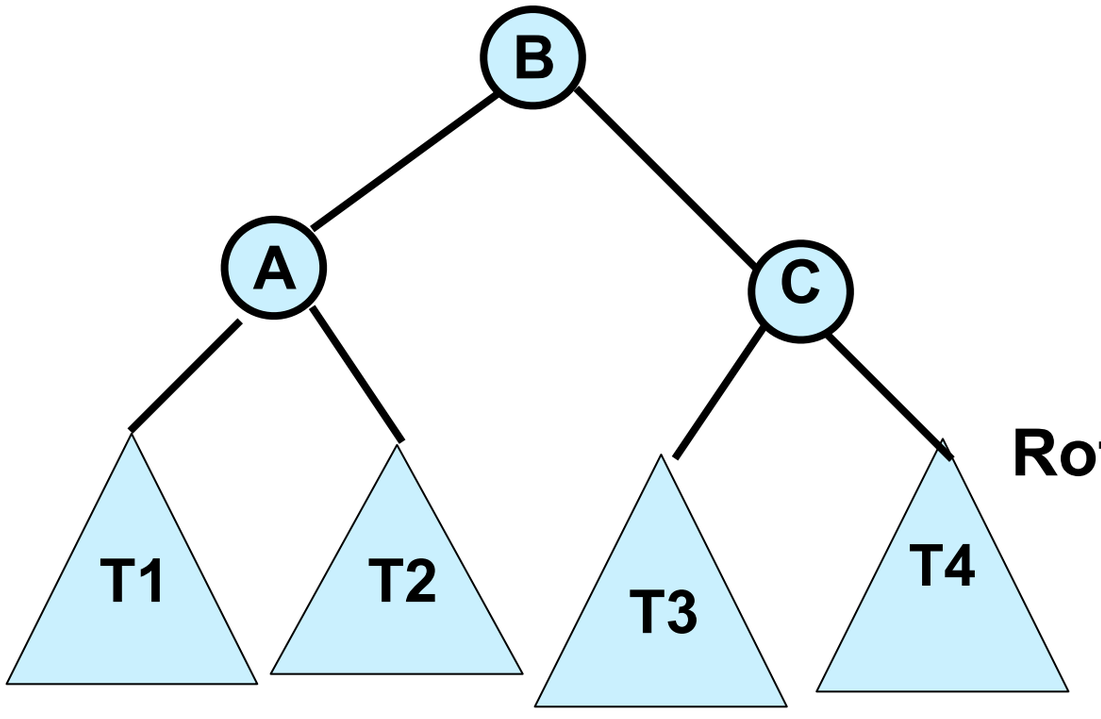
La doppia rotazione



Rotazione a sinistra su A

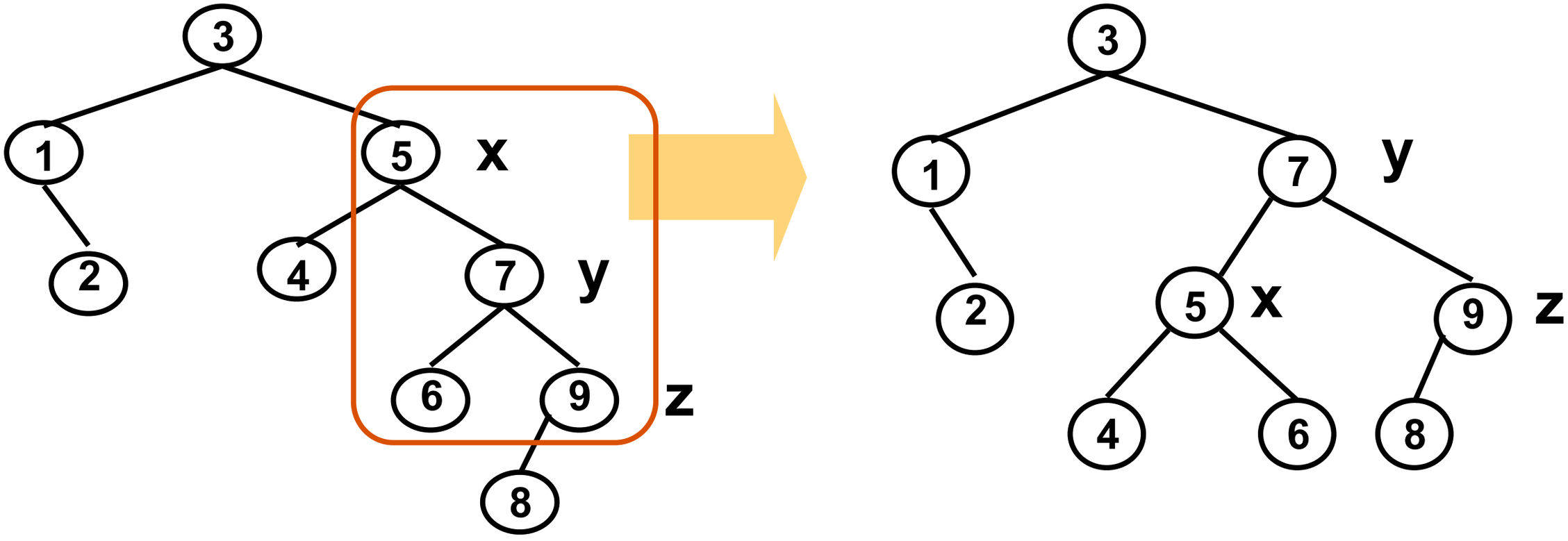
Si rientra nel caso precedente eliminando lo spigolo, o cammino zig-zag



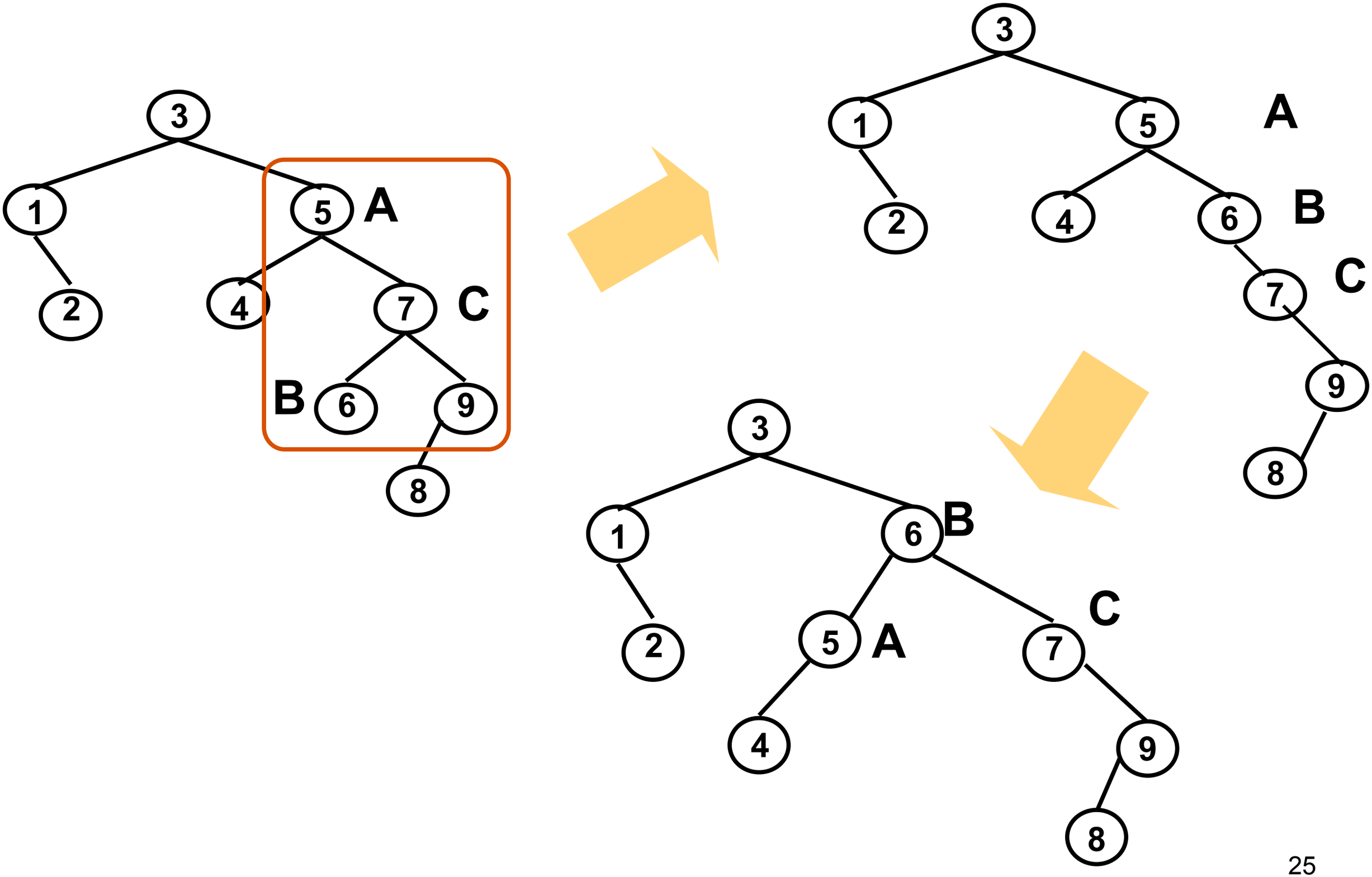


Rotazione a destra su C

Esempio di rotazione a sinistra su 5



Esempio di doppia rotazione a sinistra su 5



Operazioni su ABR in O(lg n)

In un ABR con n nodi e di altezza h le operazioni di

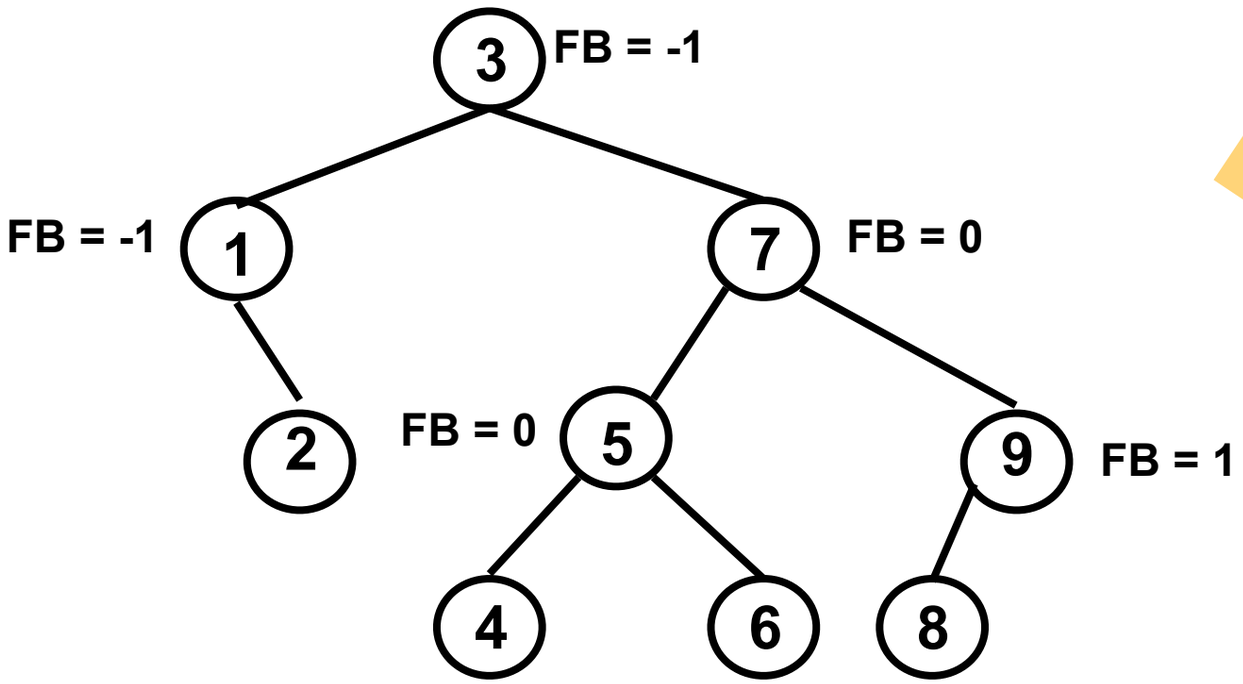
- **ricerca**
- **inserimento**
- **cancellazione**

hanno una complessità asintotica O(h),

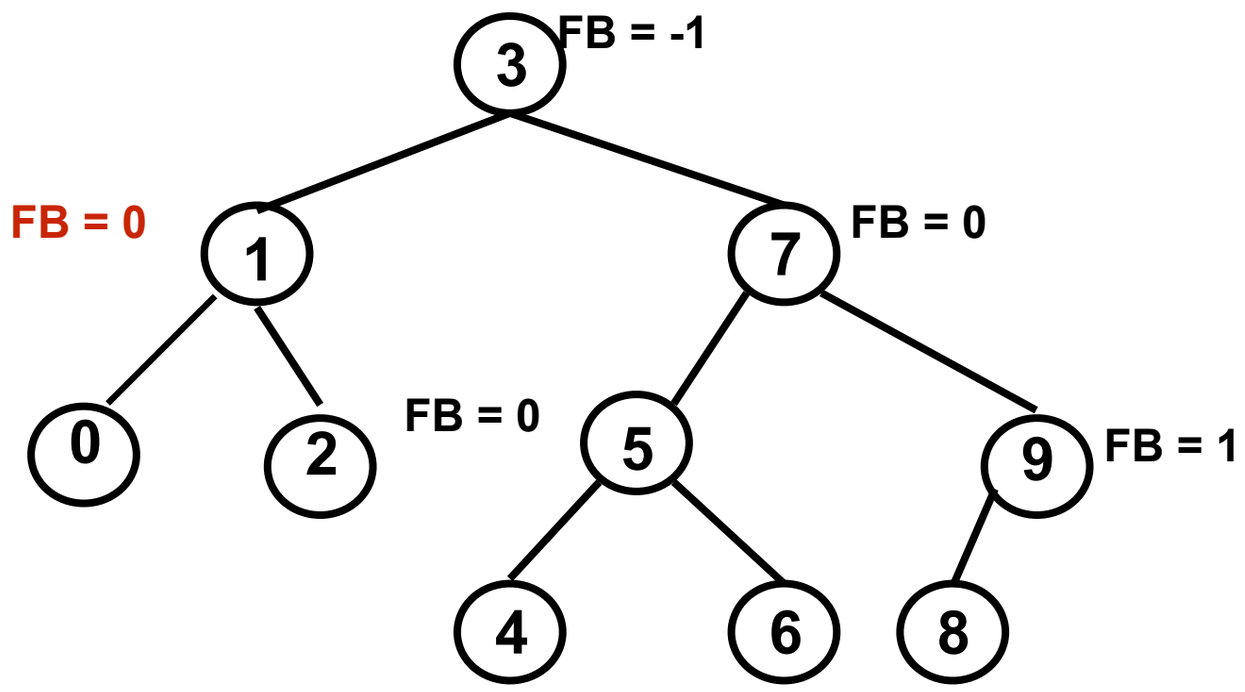
con lg n ≤ h < n

in un albero bilanciato in altezza dimostreremo che hanno una complessità asintotica O(lg n), perchè l'altezza è O(lg n) e il ribilanciamento dell'albero può essere fatto in O(lg n)

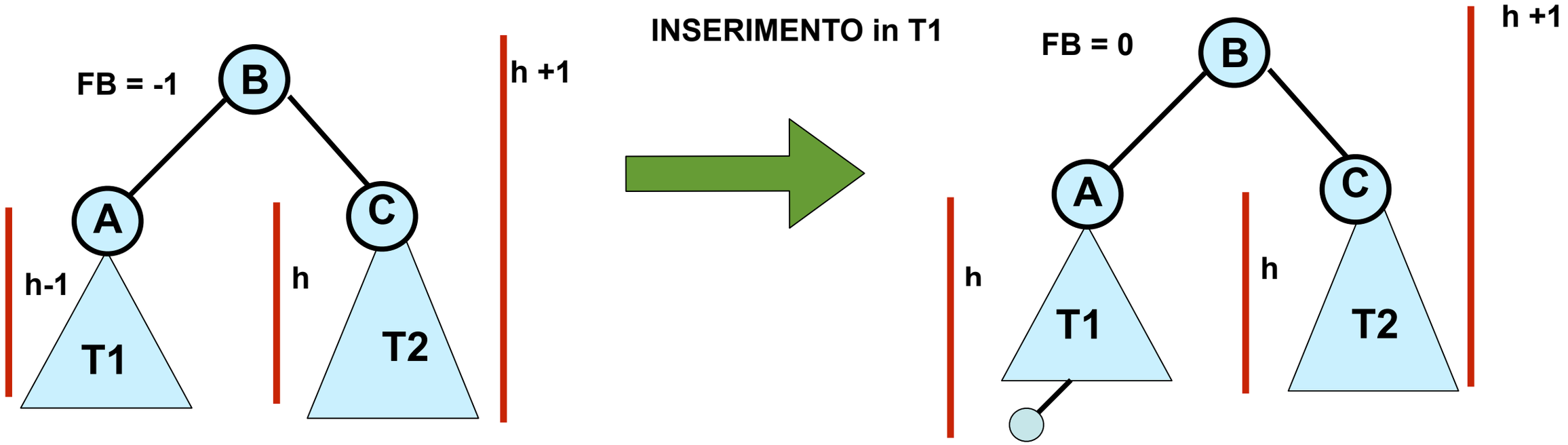
Esempio inserimento in AVL



Inserimento di 0

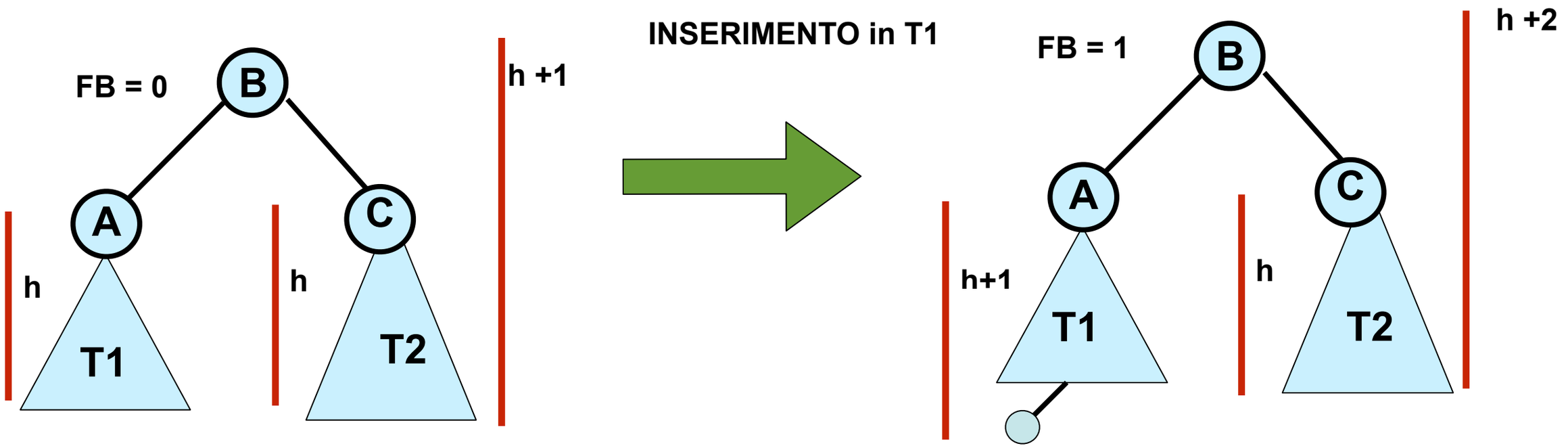


INSERIMENTO: aggiornamento FB



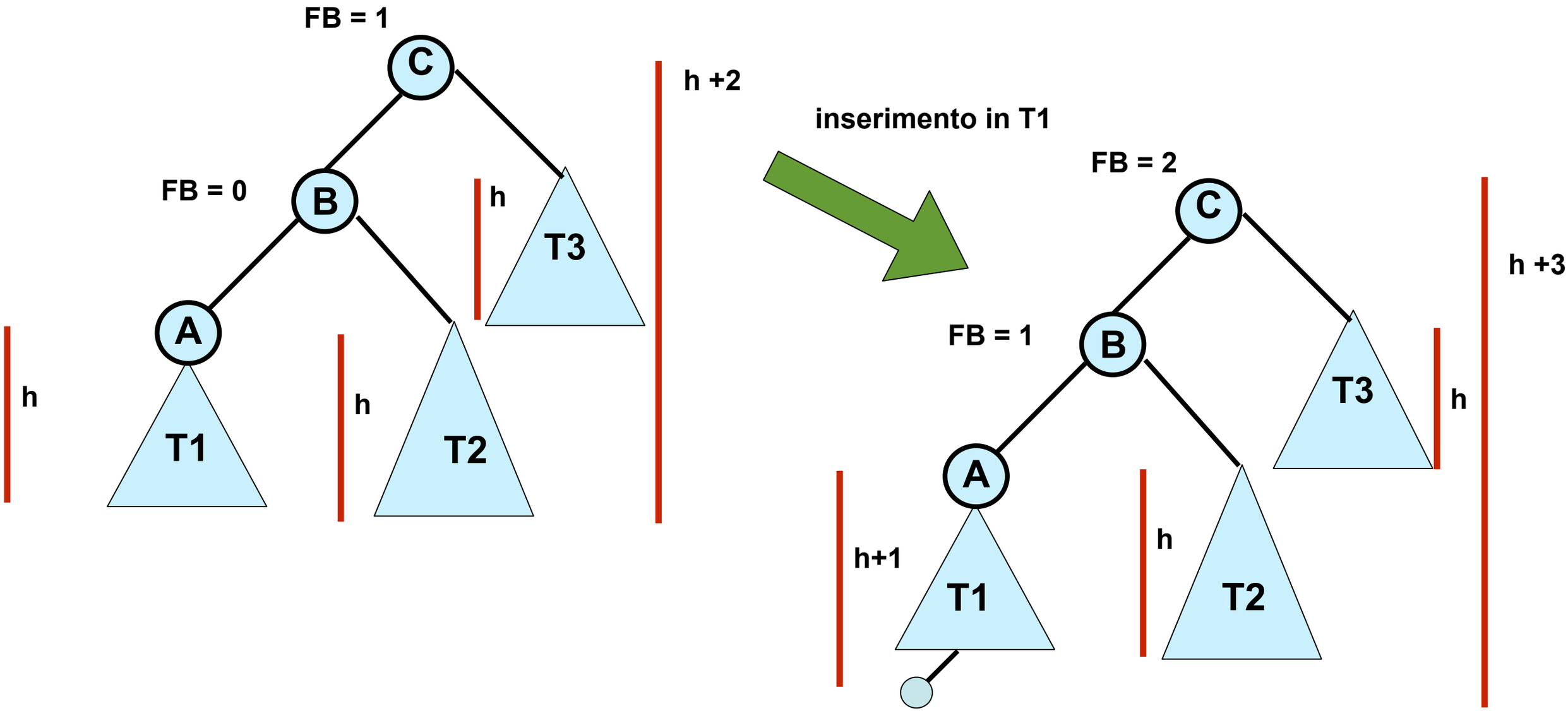
Risalendo verso la radice il fattore di bilanciamento passa da -1 a 0 quindi l'inserimento ha aumentato l'altezza del sottoalbero sinistro che però era più basso di 1, quindi complessivamente l'altezza del sottoalbero radicato in B non cambia e si può fermare il processo di aggiornamento dei FB.
Il caso simmetrico è analogo.

INSERIMENTO: aggiornamento FB



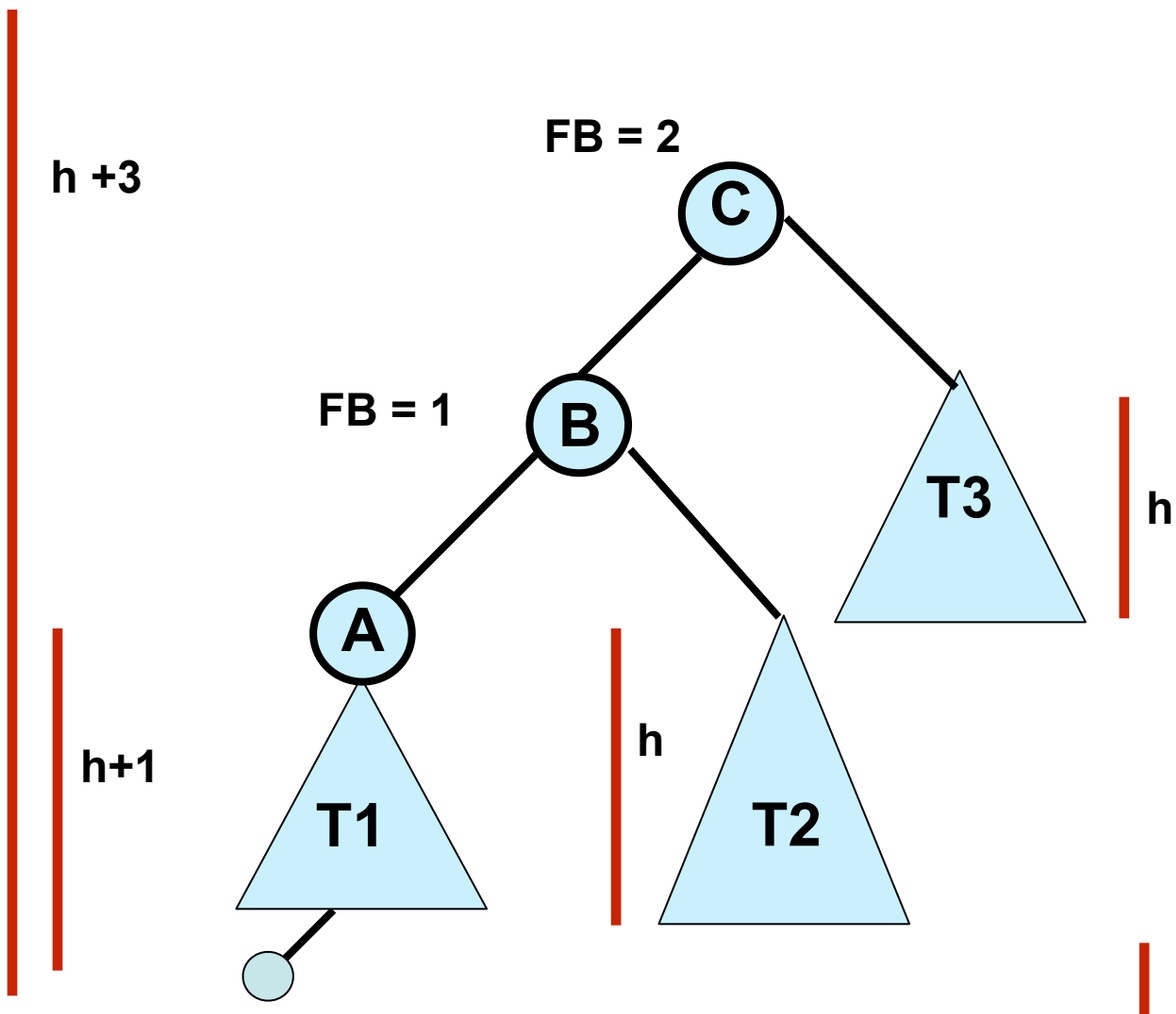
Risalendo verso la radice il fattore di bilanciamento passa da 0 a 1 quindi il sottoalbero sinistro ha aumentato la sua altezza e complessivamente tutto il sottoalbero radicato in B, quindi si deve controllare il padre di B.
Per esempio se si inserisce come foglia più a sinistra una chiave in un albero completo si dovrà cambiare i FB di tutti i nodi sul cammino più a sinistra portandoli da 0 a 1.
Naturalmente il discorso è analogo se FB passa da 0 a -1.

Inserimento: CASO LEFT-LEFT

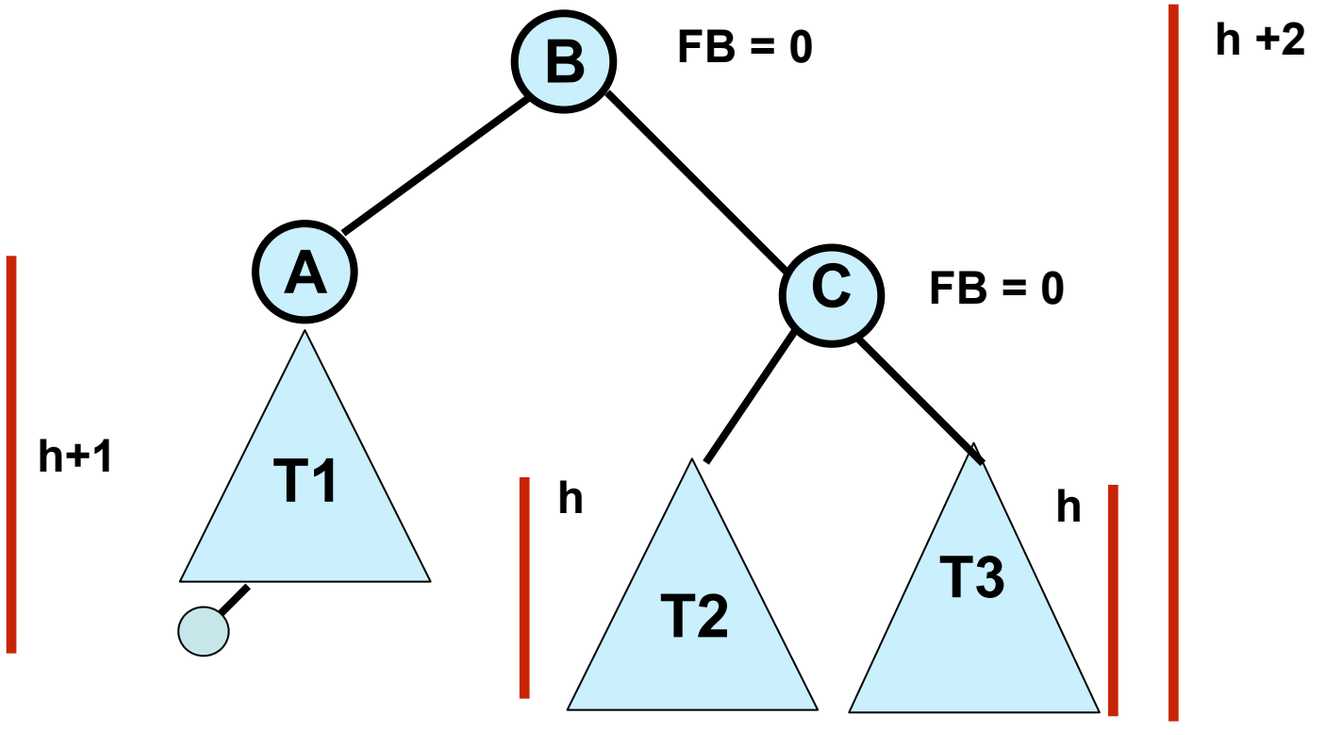


L’inserimento nel sottoalbero radicato in A può provocare un aumento di altezza nel sottoalbero sinistro di C, nodo nel quale il fattore di bilanciamento diventa illegale.

CASO LEFT-LEFT

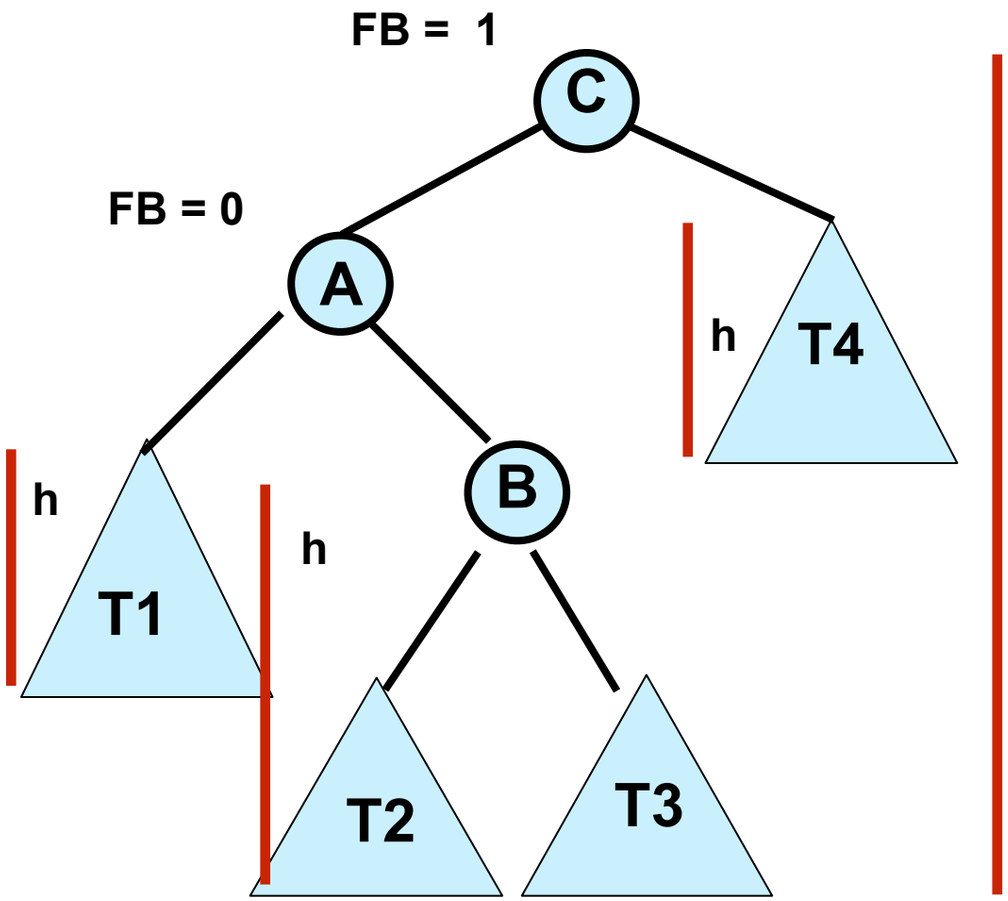


Rotazione a destra su C

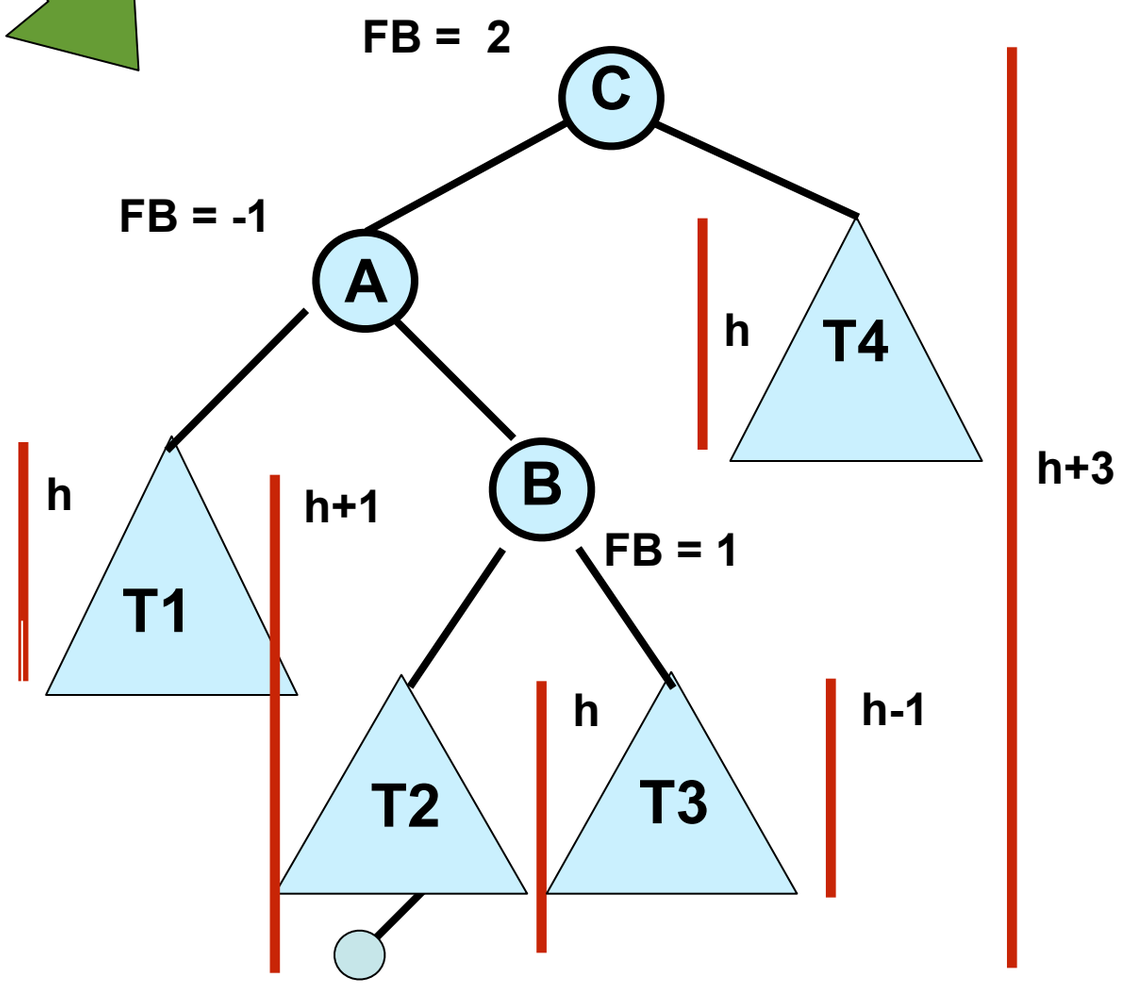


Dopo la rotazione l'altezza di B è quella del nodo C prima dell'inserimento, quindi non è necessario risalire ancora verso la radice per ribilanciare l'albero!

CASO LEFT-RIGHT

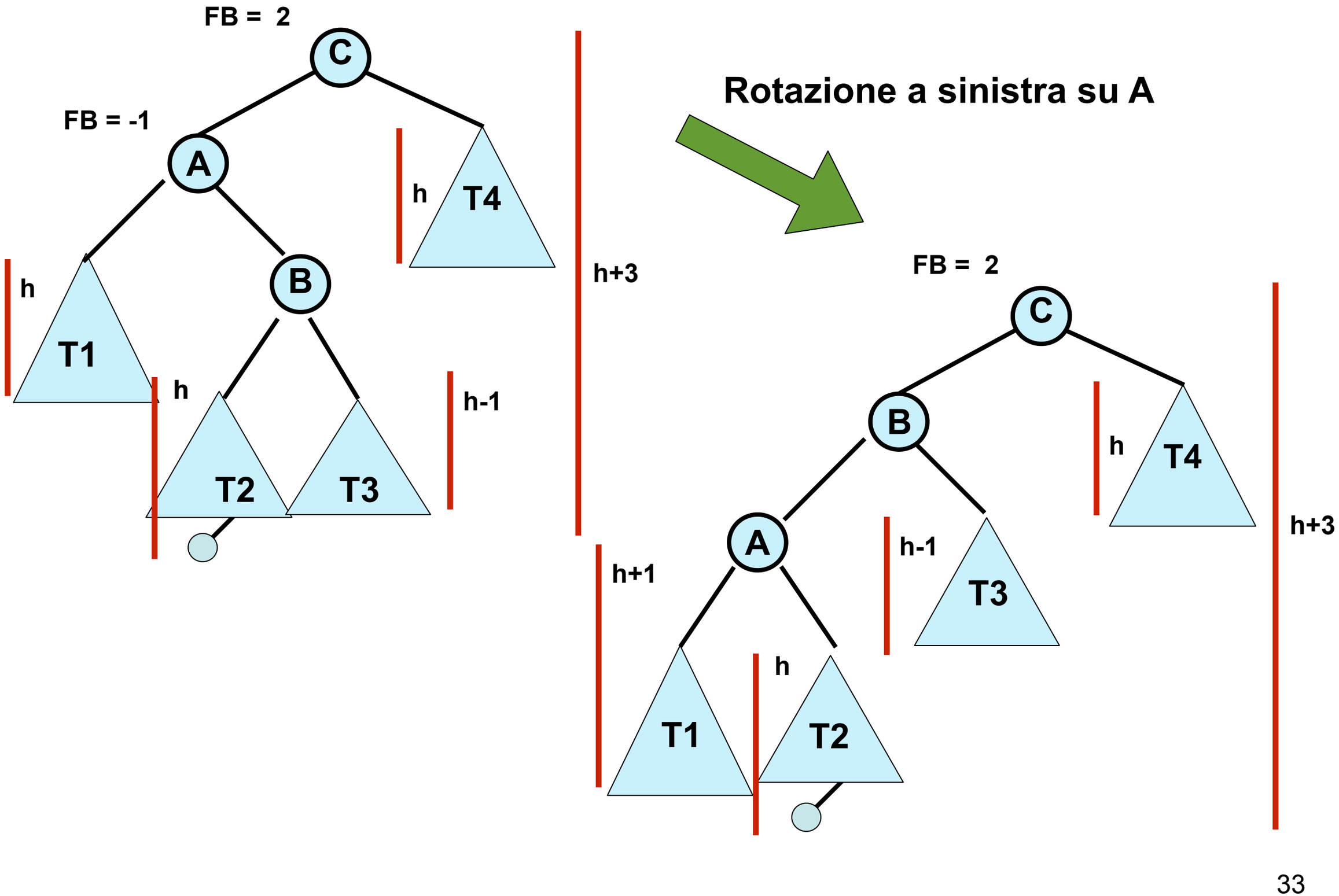


inserimento in T2

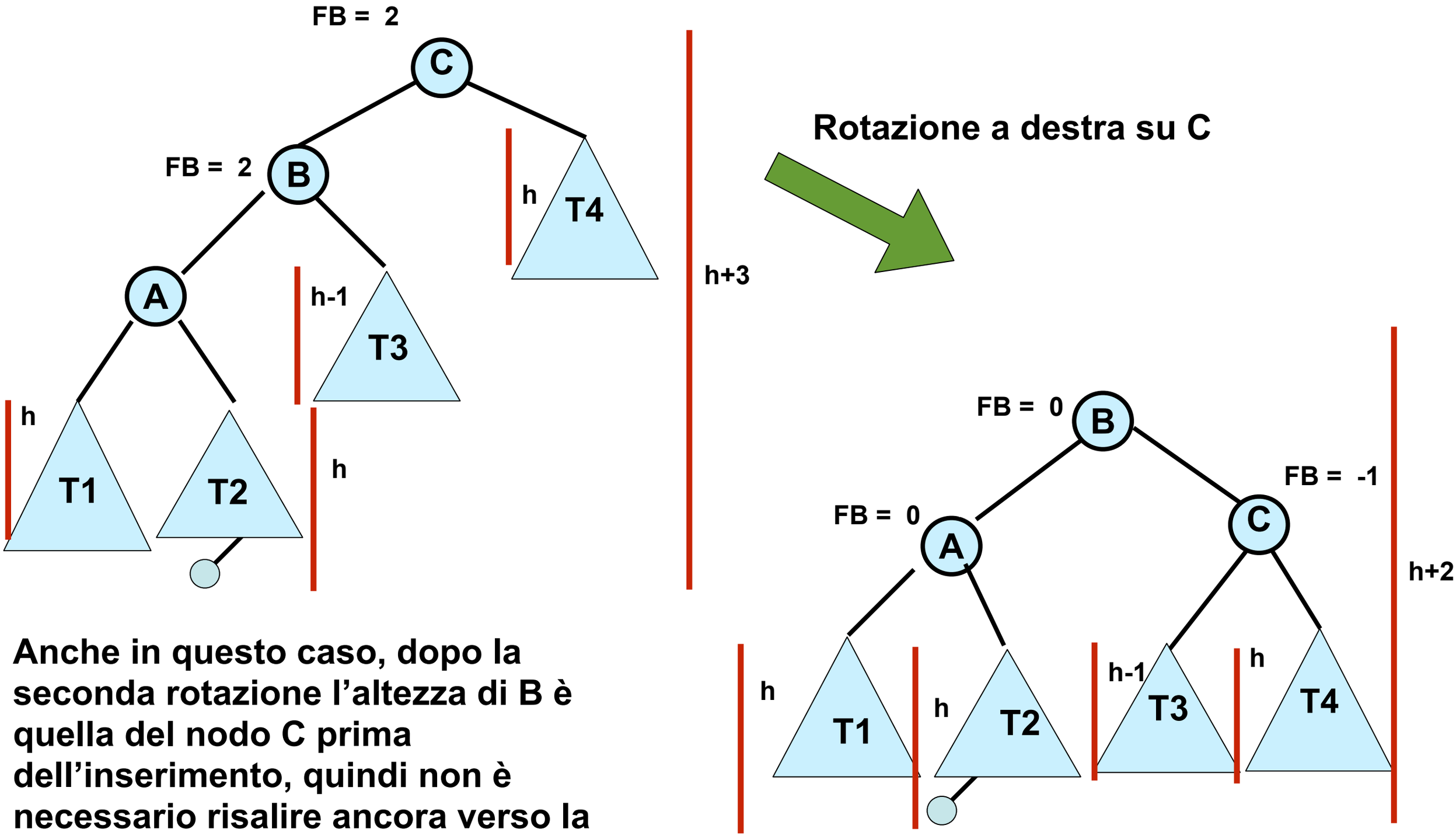


L’inserimento nel sottoalbero radicato in B può provocare un aumento di altezza nel sottoalbero sinistro di C, nodo nel quale il fattore di bilanciamento diventa illegale.

LA PRIMA ROTAZIONE



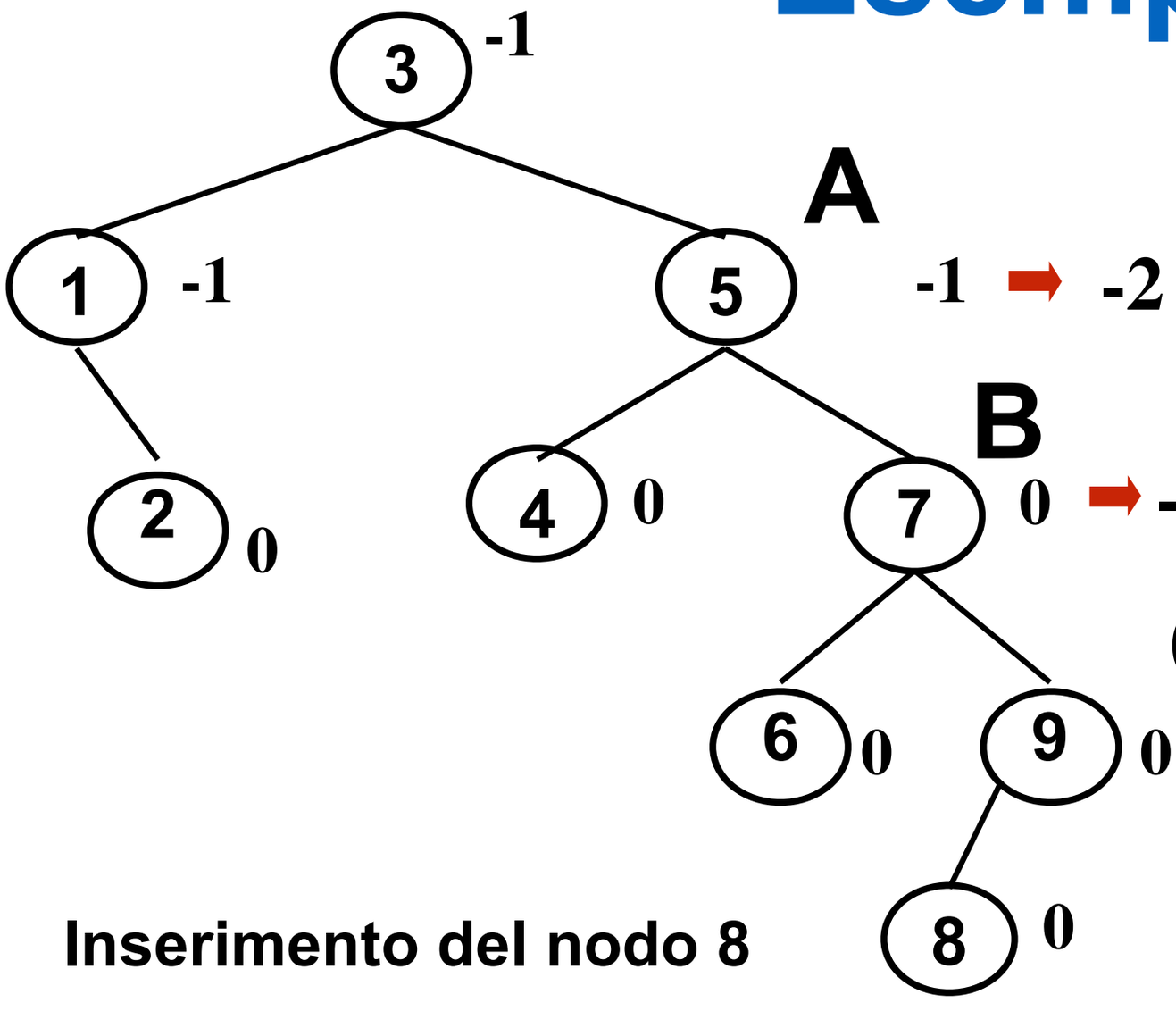
La seconda rotazione



Anche in questo caso, dopo la seconda rotazione l'altezza di B è quella del nodo C prima dell'inserimento, quindi non è necessario risalire ancora verso la radice per ribilanciare l'albero!

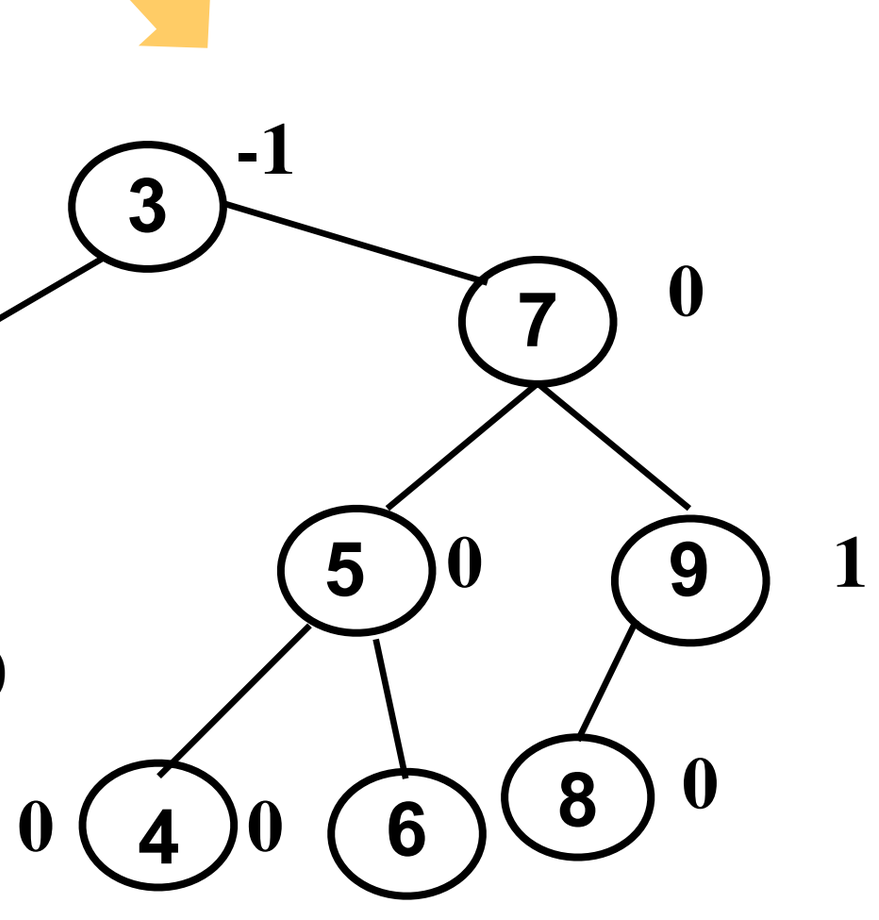
Esempio inserimento

caso right-right

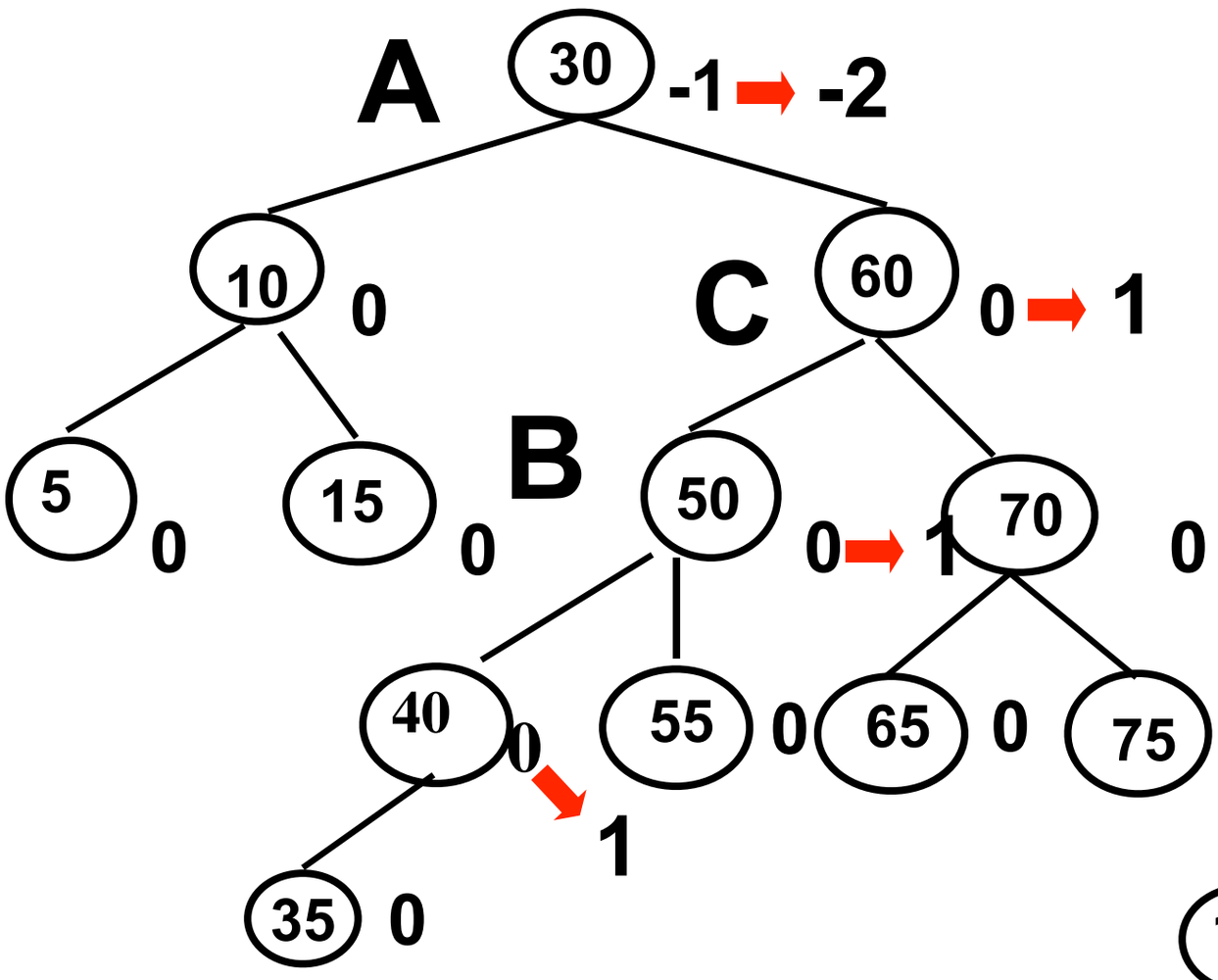


Rotazione a sinistra su A

Inserimento del nodo 8

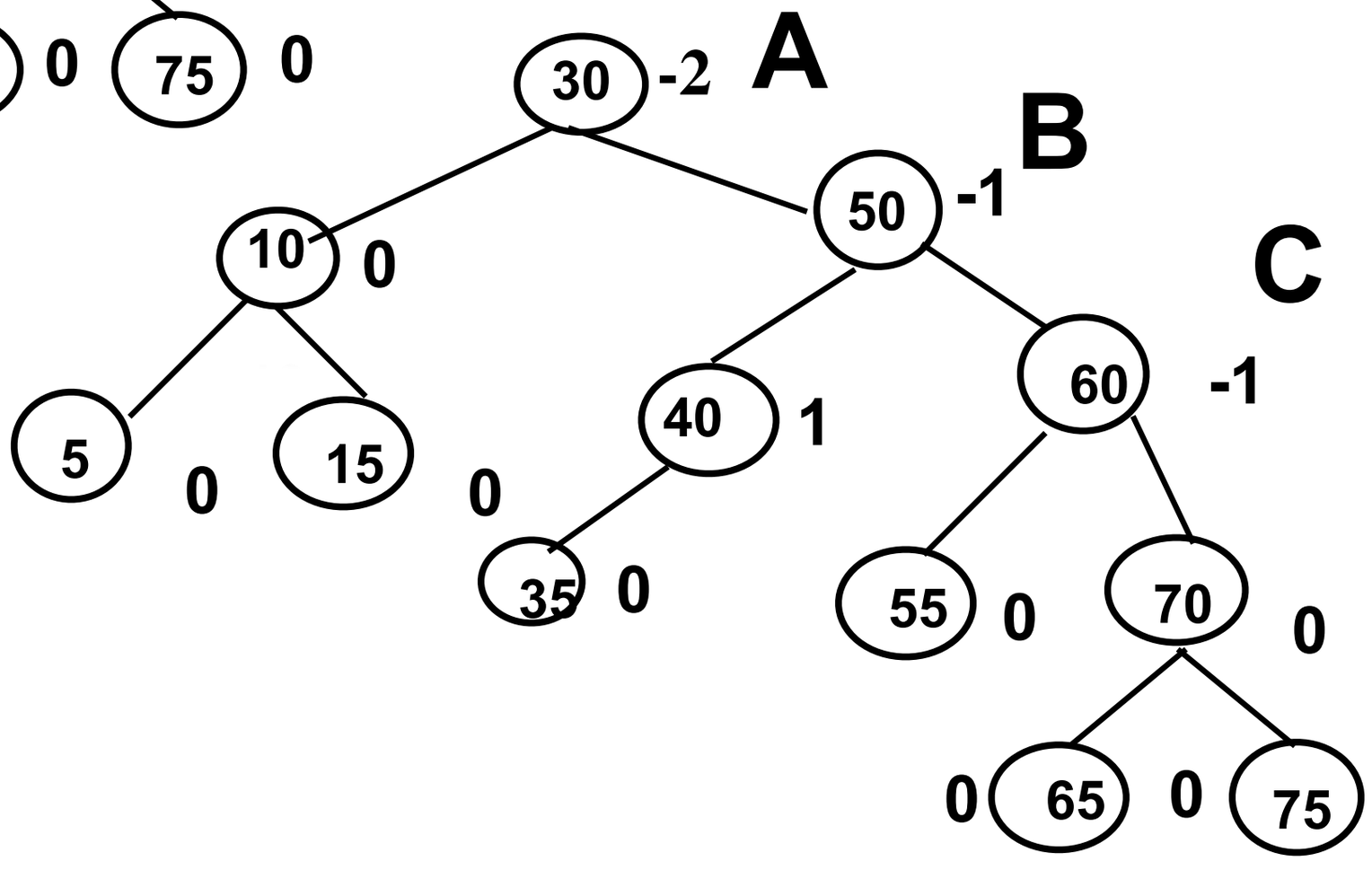


Esempio inserimento



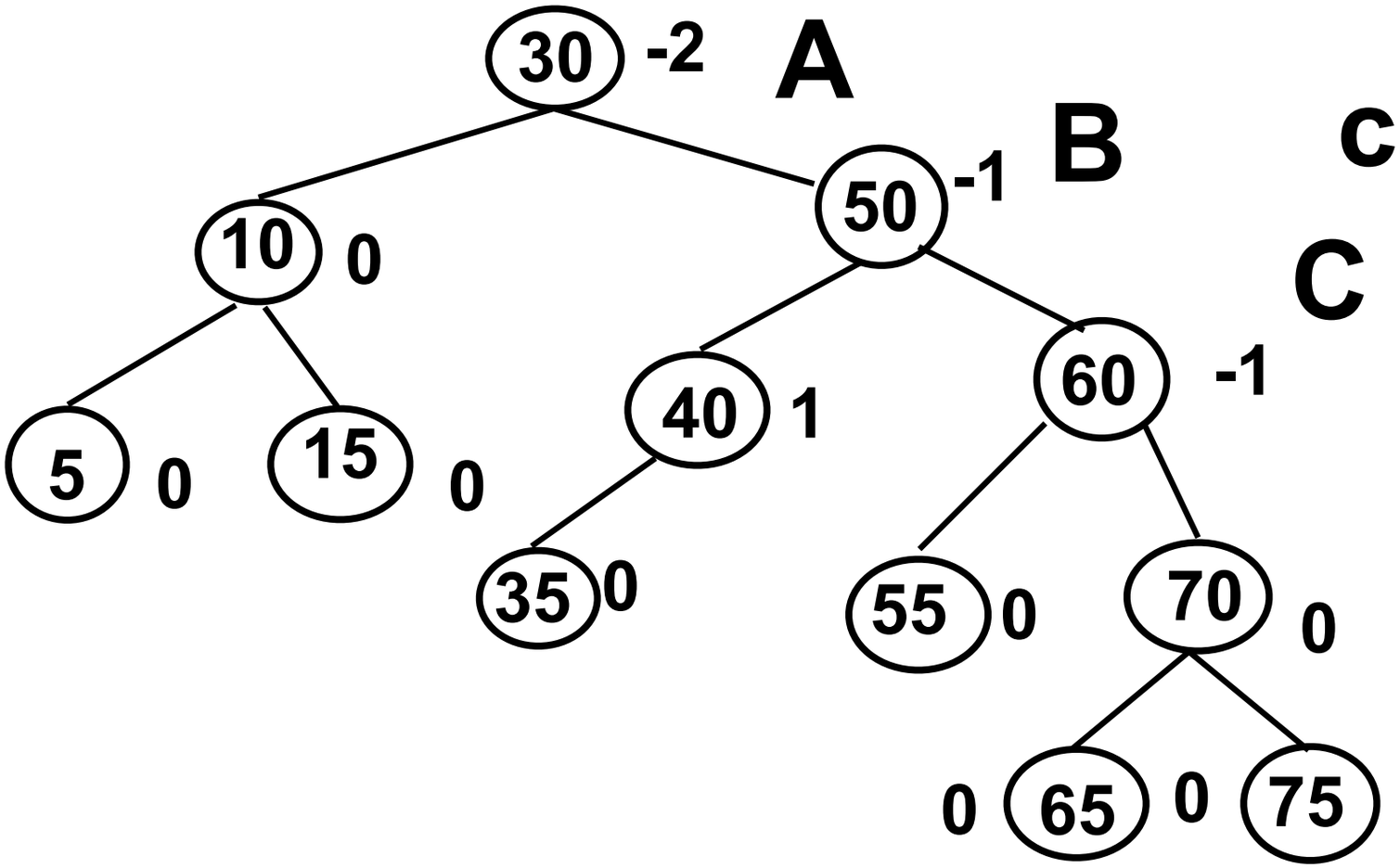
caso RIGHT- LEFT

Rotazione a destra su C

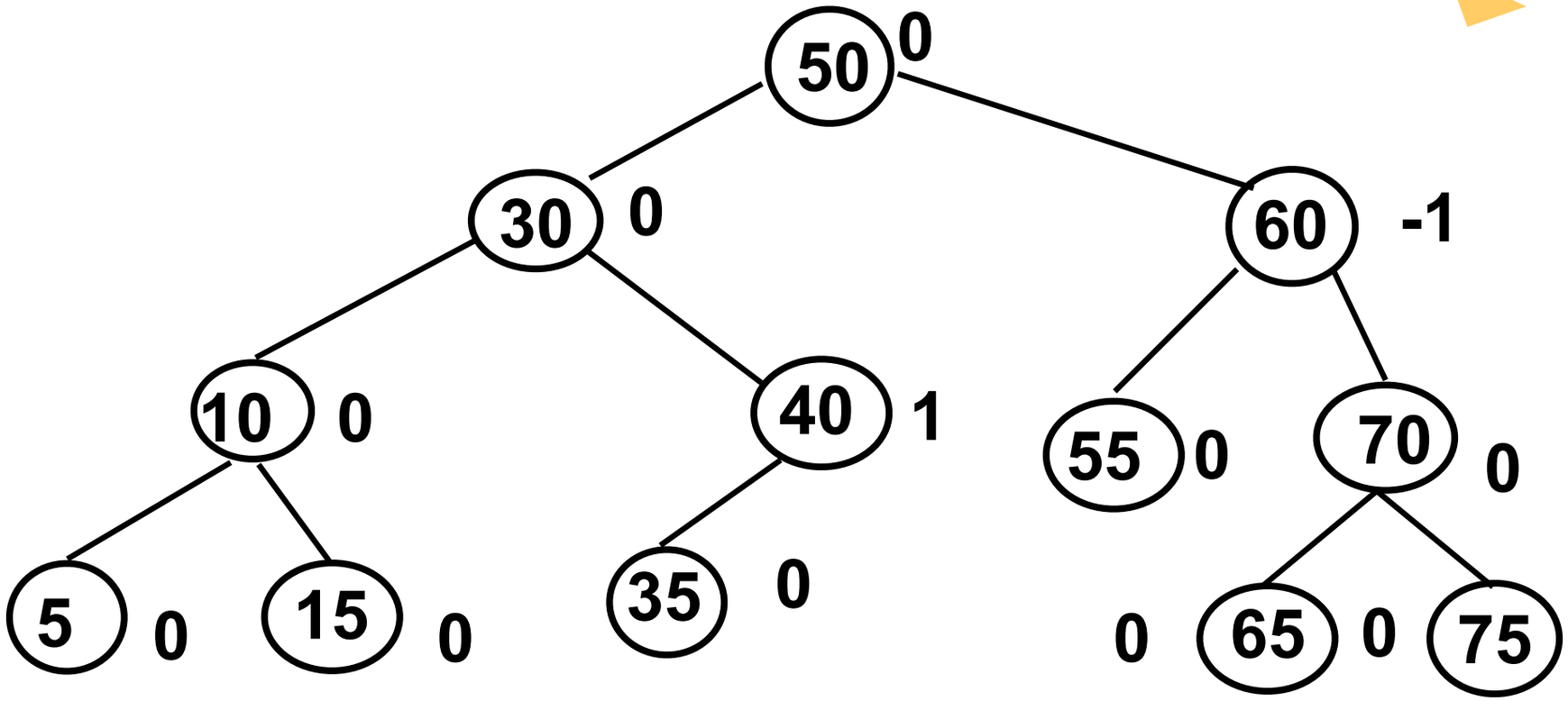


Inserimento del nodo 35

caso RIGHT- LEFT



Rotazione a sinistra su 30



Complessità

L’ inserimento da solo prende un tempo O(lg n).

Risalendo verso la radice si aggiornano i fattori di bilanciamento nei nodi e se un fattore diventa illegale (-2 o 2) con al più due rotazioni si ripristina la proprietà del bilanciamento in altezza.

Quindi anche le operazioni di ripristino delle proprietà di bilanciamento hanno un costo O(lg n).