DIZIONARIO

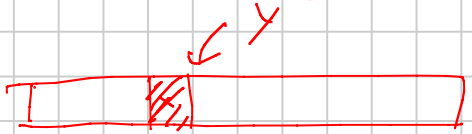
Titolo nota 17/03/2016

Alberi binari di ricerca ABR

ricerca	$O(h)$	$O(n)$
inserzione	$O(h)$	$O(n)$
min e max	$O(h)$	$O(n)$
pred e succ	$O(h)$	$O(n)$
ordinamento = Visita in ordine simmetrico		$\Theta(n)$
cancellazione	$O(h)$	$O(n)$

In media costano $O(\log n)$

Se l'albero è abbastanza bilanciato funziona bene



Ricerca di x $\Theta(\log n)$
completare l'array $O(n)$

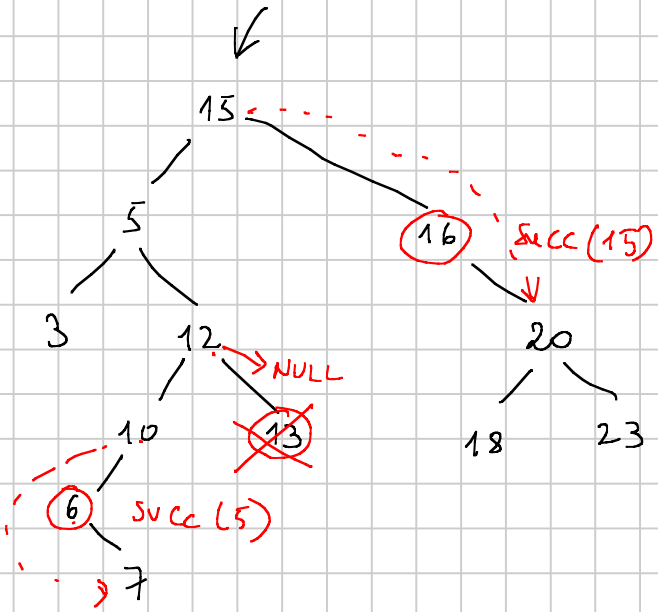
Inserzione $\Theta(\log n)$
spostamento dentro di tutti i successori $O(n)$

Ricerca Binaria in array va bene per un insieme statico

ABR

Si cerca in T
il nodo successore di
x

Il successore e il predecessore sono gli unici nodi che possono sostituire x.



- il successore è il minimo del sottoalbero destro di x
- il successore ha figlio sinistro = ∅

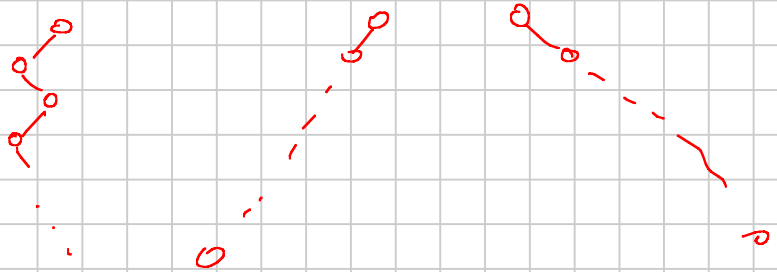
cancella (T, x);

cancella (T, 13) se foglia facile
cancella (T, 16) facile bypass
cancella (T, 15)
cancella (T, 5)
} "difficile"

1) determinare il successore che è un nodo facile da cancellare
2) Eliminare il succ
3) Sostituiscilo alla radice

Titolo nota 17/03/2016

ABR Alberi degeneri



$\Rightarrow O(n)$

alberi binari di ricerca "abbastanza" bilanciati perché mantenga $h = O(\log n)$ e inoltre mantenere l'invariante costo $O(\log n)$

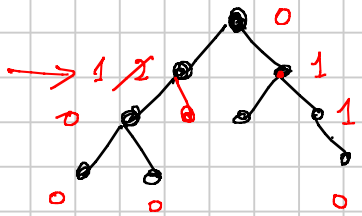
alberi AVL
alberi rosso-neri (Cormen, ..) } bilanciati in altezza
alberi weight-balanced bilanciati sul numero nodi
alberi 2-3 (versione in memoria centrale dei B-alberi)

ABR: alberi AVL (Adelson Velsky – Landis)

alberi bilanciati in altezza

$\Rightarrow$ $\forall$ nodo u di un AVL $\Rightarrow$ $|h(u.sx) - h(u.dx)| \leq 1$

un albero perfettamente bilanciato è un AVL.



NON È AVL

COL NODO ROSSO SÌ

abbastanza bilanciato da garantire altezza max $O(\log n)$

Th. Un albero AVL ha un altezza $h \in O(\log n)$

Titolo nota 17/03/2016

Prova

Quali sono gli alberi più sbilanciati possibile nella famiglia degli AVL.

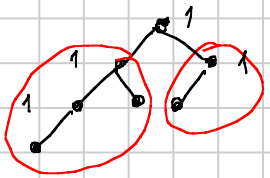
Fissiamo h e definiamo i più sbilanciati.

$h = 0$

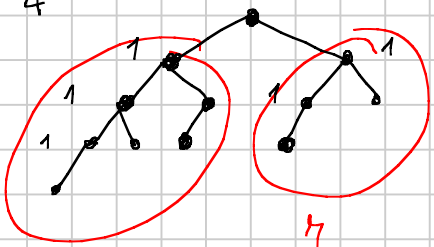
$h = 1$

$h = 2$

$h = 3$



$h = 4$



F_h

F_{h-2}

F_{h-1}

alberi di Fibonacci: rappresentano l'albero AVL di altezza h più sbilanciato possibile

n_h = numero di nodi in un albero di Fibonacci di altezza h

$$\begin{cases} n_0 = 1 \quad n_1 = 2 \\ n_h = 1 + n_{h-1} + n_{h-2} \end{cases}$$

CASO PESSIMO DEGLI AVL

h	0	1	2	3	4	5	6	7	8	9	10
FIB	0	1	1	2	3	5	8	13	21	34	55
n_h di T_h	1	2	4	7	12	20	33	54			

$$\boxed{n_h = FIB_{h+3} - 1}$$

si prova per induzione

$$FIB_h = \frac{\phi^h - (1-\phi)^h}{\sqrt{5}} \qquad \phi = \frac{1+\sqrt{5}}{2}$$

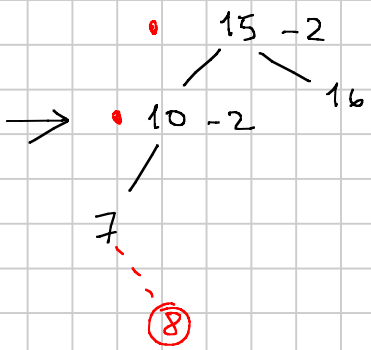
$$FIB_h \geq \frac{\phi^h - 1}{\sqrt{5}} \geq c^h, \; c > 1 \text{ e } h > 2$$

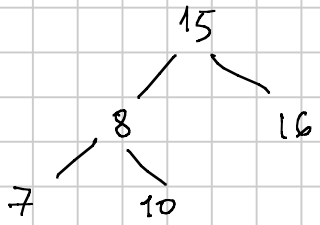
$$n_h \geq c^h \qquad h \leq \log(n_h) \Rightarrow h = O(\log n_h) \Rightarrow h = O(\log n)$$

$$h \leq 1.44 \log n + c$$

Come mantenere l'invariante ?

Titolo nota 17/03/2016





10 e 15 si sbilanciano

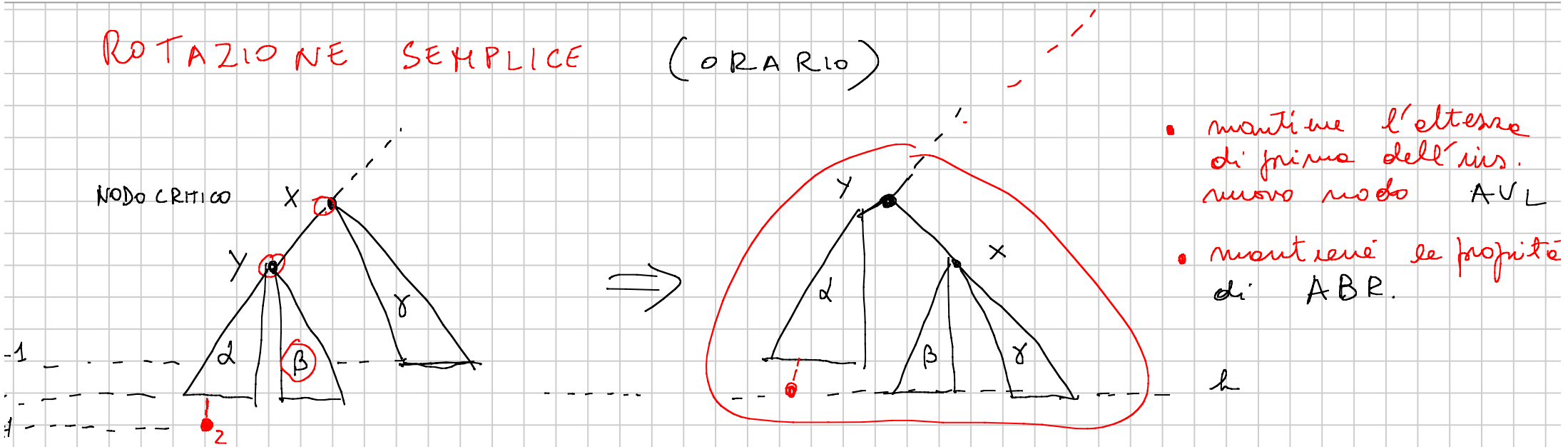
selezione NODO CRITICO = 10

è il nodo a profondità massima che si è sbilanciato

ROTAZIONI

Ribilanciando il nodo critico $\Theta(1)$ ribilancio l'intero albero.

ROTAZIONE SEMPLICE (ORARIO)

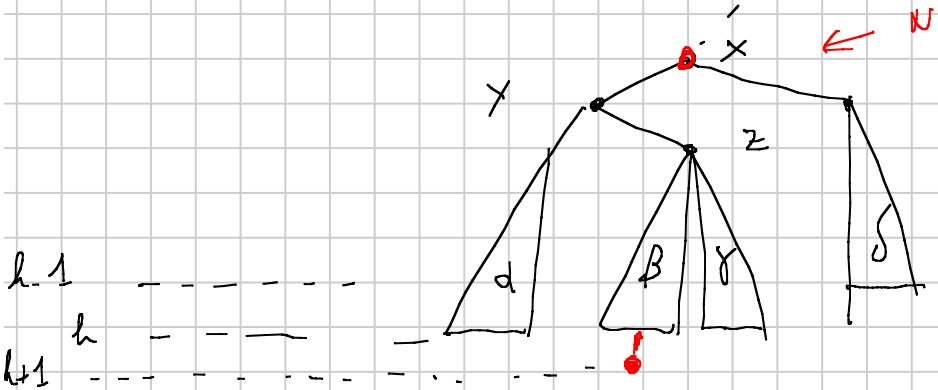


- mantiene l'altezza di prima dell'ins. nuovo nodo AVL
- mantiene le proprietà di ABR.

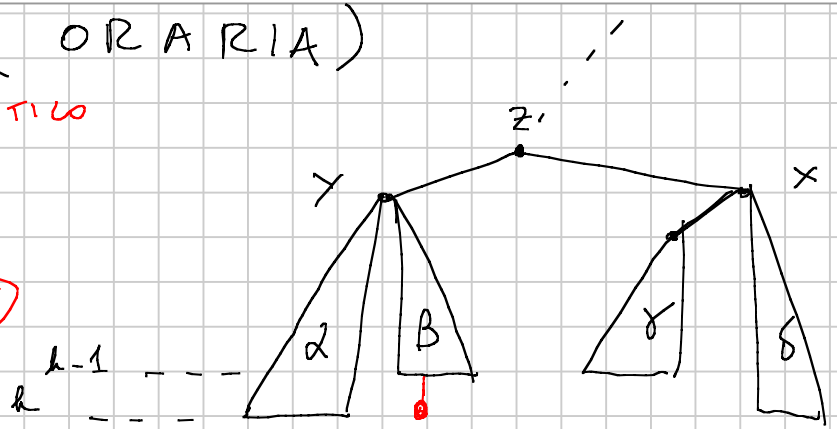
È ripristinata l'altezza di prima dell'inserimento di z.

Viene condotto un numero costante di confronti $\Theta(1)$

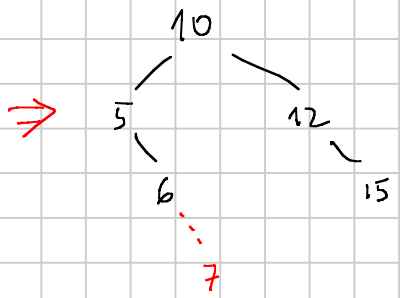
ROTAZIONE DOPPIA (ORARIA)

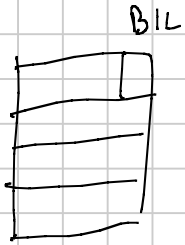


$\Theta(1)$



mantiene altezza h
(come prima dell'inserimento)
mantiene ABR





m. bil 2 bit

-1, 0, +1

ROTAZ. SEMPLICE ANTIORARIA