

## INSERIMENTO IN ALBERI RB

Quando inseriamo un nodo dobbiamo assegnargli un colore.

Conviene averlo rosso perché inserendolo nero modificherebbe l'ulteriore nera, violando sicuramente il 4° vincolo degli alberi RB.

Tuttavia essendo rosso potrebbe violare la 3<sup>a</sup> proprietà (tutti i figli di rossi sono neri).

### INSERT-RB( $T, k$ )

IF  $T \neq \text{NIL}(T)$

Se non inseriamo sotto una foglia

IF  $T \rightarrow \text{key} < k$  THEN

$T \rightarrow dx = \text{INSERT-RB}(T \rightarrow dx, k)$

$T = \text{BILANCIARB-DX}(T)$

Dobbiamo bilanciare controllando eventuali violazioni.

ELSE

IF  $T \rightarrow \text{key} > k$  THEN

Se abbiamo una violazione, sara' sempre unica e sul vincolo 3, gli altri vincoli non possono essere violati post-inserimento

$T \rightarrow sx = \text{INSERT-RB}(T \rightarrow sx, k)$

$T = \text{BILANCIARB-SX}(T)$

ELSE

$T = \text{AlloraNodo}$

$T \rightarrow \text{Key} = k$

$T \rightarrow \text{color} = \text{red}$

## VIOLAZIONI IN ALBERI RB POST INSERIMENTO

Se c'è una violazione, puo' essere solo nell'albero dove andiamo a inserire.

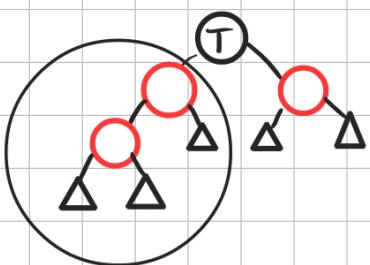
Se tuffiammo a sinistra, controlliamo se alla sinistra di T ci sono due nodi rossi consecutivi.

Il figlio rosso (che abbiamo inserito) del rosso puo' essere destro o sinistro, e sare' per forza l'unico che veda (puo' prima non c'erano violazioni).

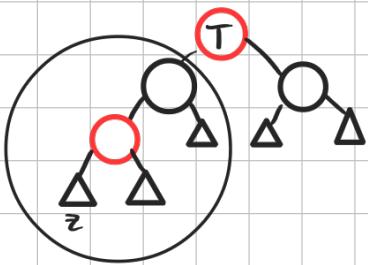
BILANCIARB controlla i figli e i nipoti dei figli veri per vedere se vedono.

Vediamo i vari casi di violazione (assumiamo un inserimento a sx)

### CASO 1



Se T ha due figli rossi possono cambiare i colori e verificare che questo cambio non genera violazioni.



Il vincolo 3 non è banalmente violato (i nodi N11 sono neri)

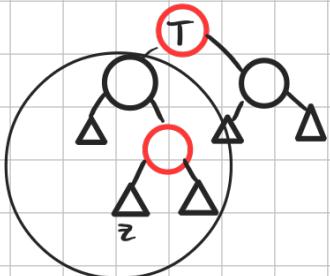
Dobbiamo controllare il 4 vincolo, ovvero confrontare la nuova alzata nera con la precedente.

È facile vedere che considerando un nodo  $\tau$  prima aveva solo un nodo nero (la radice) e ora ne ha sempre 1 (quello cambiato di colore). Analogamente vale nel sottoalbero destro.

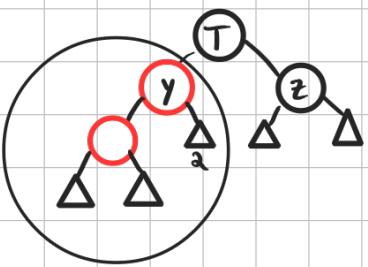
Potrebbe succedere che  $T$  abbia padre rosso, ma finché è solo una riduzione se ne occuperanno le chiamate ricorsive (eventualmente cambiandogli colore).

Se  $T$  è radice globale alla fine lo riponiamo nero (la radice è sempre nera per comodità, altrimenti avremmo problemi con l'algoritmo).

Volgono gli stessi ragionamenti in questo caso:

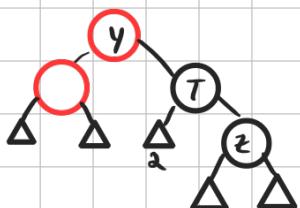


## CASO 2



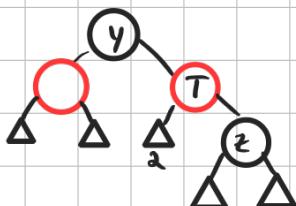
Non possiamo agire come nel caso 1 perché abbiamo una violazione a destra.

Possiamo eseguire una rotazione



Adesso però abbiamo una violazione su entrambi i violati 3 e 4.

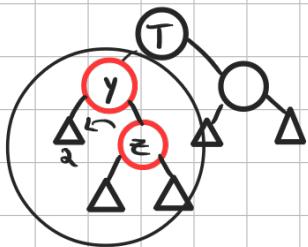
Possiamo però cambiare colore e risolvere.



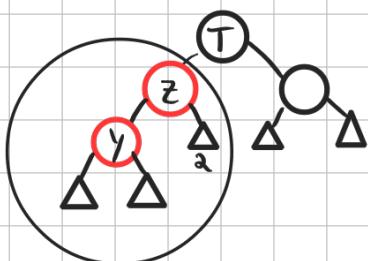
Ora abbiamo di nuovo le altre nere connette e i figli dei rossi sono neri.

Non vale lo stesso ragionamento per il caso speculare  
(come negli AVL). Vediamo quindi il caso 3.

### CASO 3



Possiamo semplicemente ruotare l'albero in  $y$  e ritrovare al caso 2.



A questo punto siamo nel caso 2.

In particolare i casi 2 e 3 sono risolutivi anche a livello globale, solo il caso 1 puo' propagare la violazione progressivamente fino alla radice

Vediamo quindi algoritmi assumendo perci' che l'albero abbia almeno altezza 2 (su altezza 1 non avremo mai violazione perch' la radice e' vera e le foglie pure).

BILANCIA RB\_SX(T)

IF  $\neg$  IS\_NIL ( $T \rightarrow s_x$ ) THEN

IF ( $T \rightarrow dx \rightarrow \text{color} = \text{red}$ ) THEN

① IF ( $T \rightarrow s_x \rightarrow \text{color} = \text{red}$ ) THEN

IF ( $T \rightarrow s_x \rightarrow s_x \rightarrow \text{color} = \text{red}$  OR

controllo del nipote

$T \rightarrow s_x \rightarrow dx \rightarrow \text{color} = \text{red}$ ) THEN

$T = \text{CASO1}(T)$

ELSE

③ IF ( $T \rightarrow s_x \rightarrow \text{color} = \text{red}$  AND  $T \rightarrow s_x \rightarrow dx \rightarrow \text{color} = \text{red}$ ) THEN

$T = \text{CASO3}(T)$

$T = \text{CASO2}(T)$

② ELSE IF ( $T \rightarrow s_x \rightarrow \text{color} = \text{red}$  AND  $T \rightarrow s_x \rightarrow s_x \rightarrow \text{color} = \text{red}$ ) THEN

$T = \text{CASO2}(T)$

RETURN T

L'algoritmo che implementa INSERT\_RB si deve ricordare di colorare la radice finale di nero per sicurezza (propagazione del caso 1 alla radice)

## CANCELLAZIONE IN ALBERI RB

I padri delle foglie sono banalmente cancellabili perché i nodi NIL sono rimovibili senza effetti collaterali. Degli altri nodi ovviamente non possiamo scegliere il colore.

Eliminando un rosso non possiamo violare alcun vincolo.

Abbiamo problemi rimuovendo un nodo nero.

In questo caso propaghiamo il colore del nodo rimosso al figlio se propaghiamo su un rosso, diventerà nero, se propaghiamo sul nero avremo un **DOPPIO NERO**, che viola il vincolo di colore.

Cercheremo tra gli antenati del nodo doppio nero dove poter inserire un nodo nero per non violare vincoli, per poi infine eliminare questo nodo temporaneo.

### ● CANC-RB(T,K)

Analogo degli AVL

IF  $\neg \text{IS-NIL}(T)$  THEN

IF  $T \rightarrow \text{Key} < K$  THEN

$T \rightarrow dx = \text{CANC-RB}(T \rightarrow dx, k)$

Dopo aver cancellato dobbiamo bilanciare

$T = \text{BILANCIA-CANC-dx}(T)$

ELSE

IF  $T \rightarrow \text{Key} > K$  THEN

$T \rightarrow S_x = \text{CANC\_RB}(T \rightarrow S_x, K)$

$T = \text{BILANCIA\_CANC\_SX}(T)$

ELSE

$T = \text{CANC\_RB\_RADICE}(T)$

RETURN T

### ● CANC\_RB\_RADICE(T)

IF ( $\nexists \text{ISNIL}(T \rightarrow S_x)$  AND  $\nexists \text{IS-NIL}(T \rightarrow d_x)$ ) THEN

caso problematico

$\text{TMP} = \text{STACCA\_MIN\_RB}(T \rightarrow d_x, T)$

dopo la chiamata, l'albero in  $T \rightarrow d_x$   
e' un RB e restituisce il min

$T \rightarrow \text{Key} = \text{TMP} \rightarrow \text{Key}$

$T = \text{BILANCIA\_CANC\_DX}(T)$

ELSE

IF  $\nexists \text{IS-NIL}(T \rightarrow S_x)$  THEN

$\text{TMP} = T$

$T = T \rightarrow S_x$

PropagateColor (TMP, T)

dealloca (TMP)

ELSE

$\text{TMP} = T$

$T = T \rightarrow dx$

PropagateColor ( $\text{TMP}, T$ )

dealloca ( $\text{TMP}$ )

RETURN  $T$

### ● STACCA\_MIN\_RB ( $T, p$ )

IF  $\neg IS\_NIL (T \rightarrow sx)$  THEN

Finché ci sono più piccoli

$\text{TMP} = STACCA\_MIN\_RB (T \rightarrow sx, +)$

$\text{NEW\_ROOT} = BILANCIA\_CANC\_SX (T)$

eventuale bilanciamento post-remozione  
che potrebbe cambiare radice

ELSE

Te' il minimo

$\text{TMP} = T$

$\text{NEW\_ROOT} = T \rightarrow dx$

Al posto di  $T$  metteremo il suo sottosalbero dx

PropagateColor ( $\text{TMP}, \text{NEW\_ROOT}$ )

IF ( $T = p \rightarrow sx$ ) THEN

$p \rightarrow sx = \text{NEW\_ROOT}$

ELSE

$p \rightarrow dx = \text{NEW\_ROOT}$

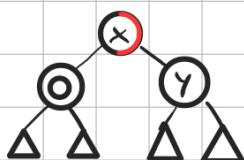
RETURN  $\text{TMP}$

$\text{TMP}$  e' il minimo che abbiamo staccato

## CASI DI BILANCIAMENTO

Vediamo che bilanciamenti effettuare dopo aver eliminato un nodo  
a sinistra e propagato il colore

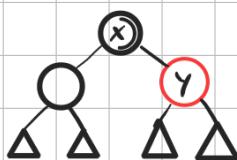
### CASO 1



Non sappiamo se  $x$  è nero o rosso.

In entrambi i casi propaghiamo il nero del doppio nero sopra.

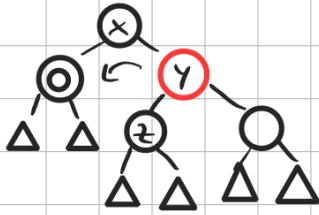
Avevamo quindi in radice un nero (se  $x$  è rosso) o un doppio nero,  
in più poniamo  $y$  rosso



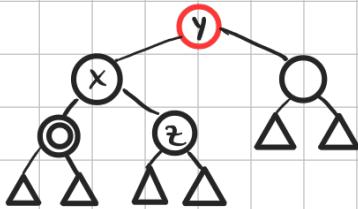
Se  $x$  era rosso (e quindi ora è nero) abbiamo ora un  
albero RB (comparandolo le altreze abbiamo restabilito le altreze iniziali)

Quindi se  $x$  era rosso il caso è risolutivo, altrimenti potrebbe  
propagare il doppio nero verso l'alto.

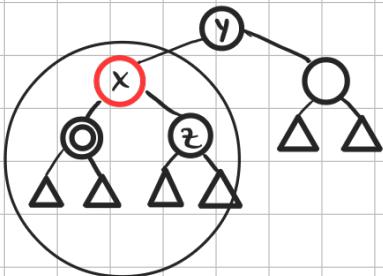
## CASO 2



Rustiamo l'albero radicato in  $x$

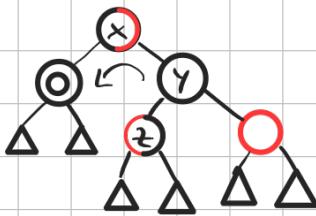


Combiniamo colori di  $x$  e  $y$  per "fare spazio" al doppio nero (che è il nostro obiettivo principale) e controlliamo le altre nere con quelle originali (ricordandoci che il doppio nero vale 2)

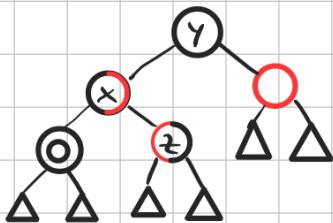


Si noti che il sottalbero  $s_x$  è proprio un caso i risolutivo, possiamo quindi richiamare il bilanciamento su  $x$  e risolvere

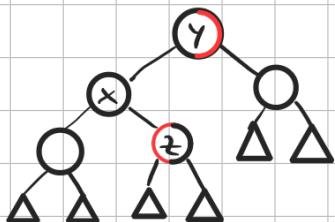
### CASO 3



Effettuiamo una rotazione sin x

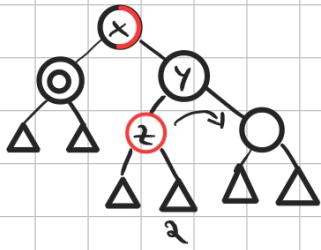


Adesso "mettiamo" i colori propagando tutto a sinistra e poi y sul figlio destro

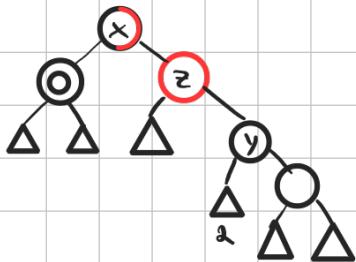


Abbiamo rimesso il doppio nero, non possiamo avere due rossi consecutivi e le altre nere corrispondono con quelle iniziali.  
Si noti che la radice ha lo stesso colore che aveva all'inizio x  
Questo caso e' quindi risolutivo

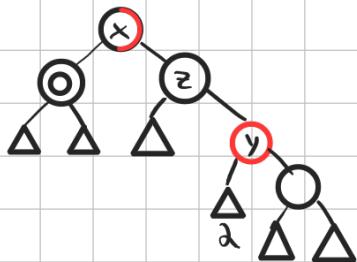
## CASO 4



Ricontrolliamo l'albero in y.



Abbriamo molteplici violazioni, ristabiliamo l'ulteriore nera scambiando di colore z e y. Siamo sicuri di poter cambiare colore a y perché ha un figlio nero e la radice di 2 è sicuramente nera (all'inizio 2 stava sotto z rosso). Questa situazione è proprio il caso 3 (iniziale) che è risolutivo



## MASSIMO NUMERO DI ROTAZIONI

Il massimo numero di rotazioni necessarie a bilanciamento è dato potenzialmente dal caso 2 si porta nel caso 4 che si porta nel caso 3, questo implica 3 rotazioni.

Negli AVL il numero massimo di rotazioni era lineare sull'altezza.

Negli alberi RB sia per inserimento che per cancellazione abbiamo un numero di rotazioni costanti.

## OSS

I casi sono studiati eliminando un nodo che ha un solo figlio perché se a due figli eliminierremo (dopo aver chiamato STACCA-MIN-RB) una " foglia".