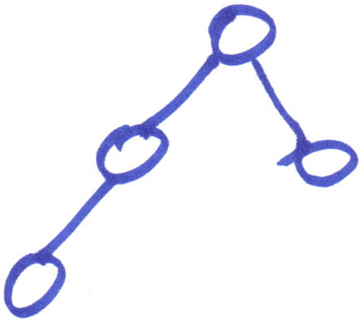
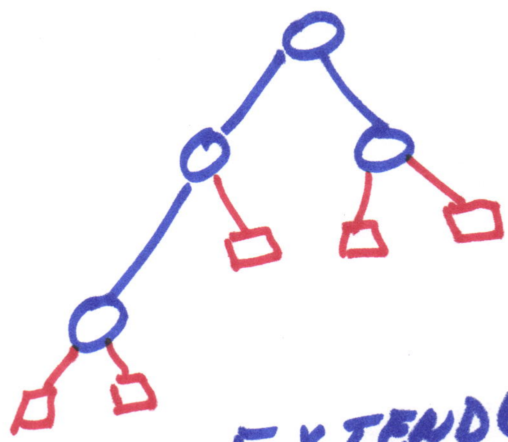
LEFTIST TREES

* LINKED BINARY TREE
* CAN DO EVERYTHING A HEAP DOES IN THE SAME TIME COMPLEXITY.
* CAN MELD (MERGE) TWO LEFTIST TREES IN $O(\log n)$ TIME.

EXTENDED BINARY TREE
(Add external nodes)



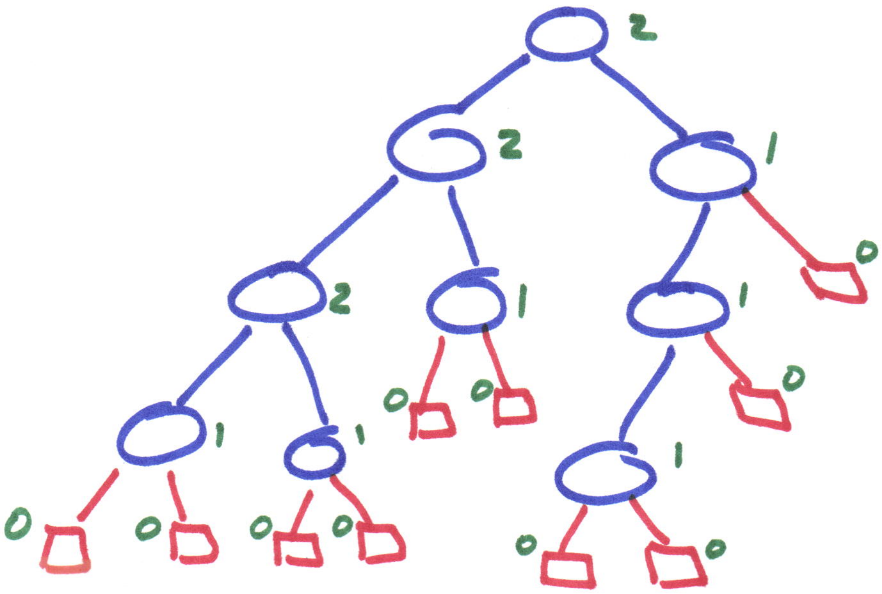
BINARY TREE



EXTENDED BINARY TREE

S( ) FUNCTION

S(X) : FOR ANY NODE X, S(X) IS THE LENGTH OF A SHORTEST PATH TO AN EXTERNAL NODE (IN THE SUBTREE ROOTED AT X).



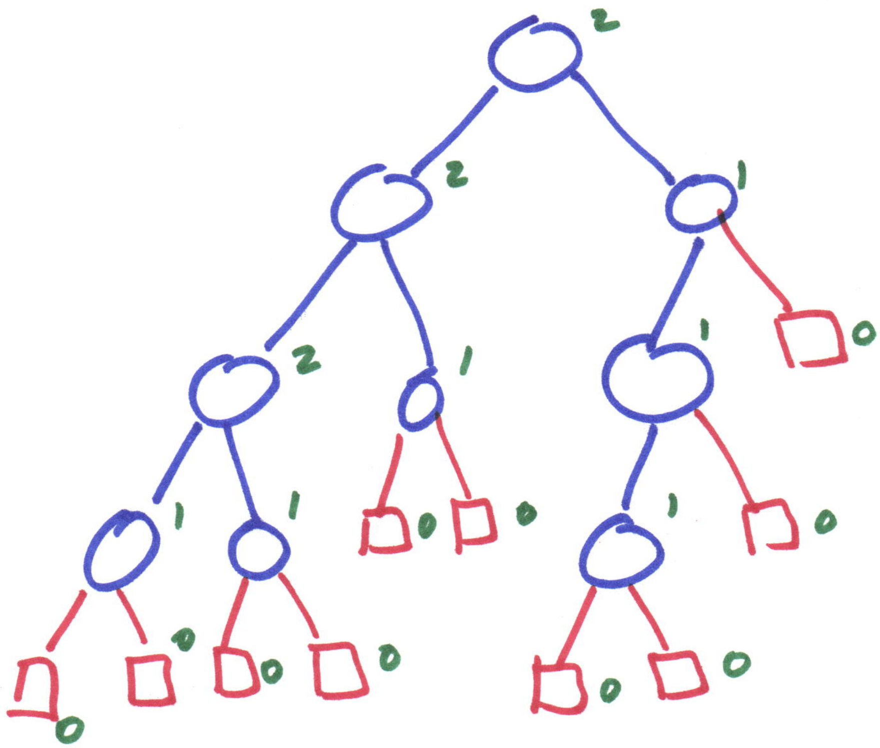
COMPUTING S(X)

$$S(x) = \begin{cases} 0 & \text{if } x \text{ is EXTERNAL NODE} \\ \min\{s(lc(x)), s(rc(x))\}+1 & \text{O.W.} \end{cases}$$

lc ... leftchild
rc ... rightchild

HEIGHT BIASED LEFTIST TREES
(HBLT)

A BINARY TREE is A HBLT
IFF
for every INTERNAL NODE X
$$s(lc(x)) \geq s(rc(x))$$



HBLT PROPERTY 1

* A SHORTEST ROOT TO EXTERNAL NODE PATH HAS LENGHT s(root). THE RIGHTMOST PATH HAS THIS LENGHT.

PROPERTY 2

* THE NUMBER OF INTERNAL NODES IS AT LEAST

$$2^{s(root)} - 1$$

(LEVEL 1.... s(root) do NOT have EXTERNAL NODES)

PROPERTY 3

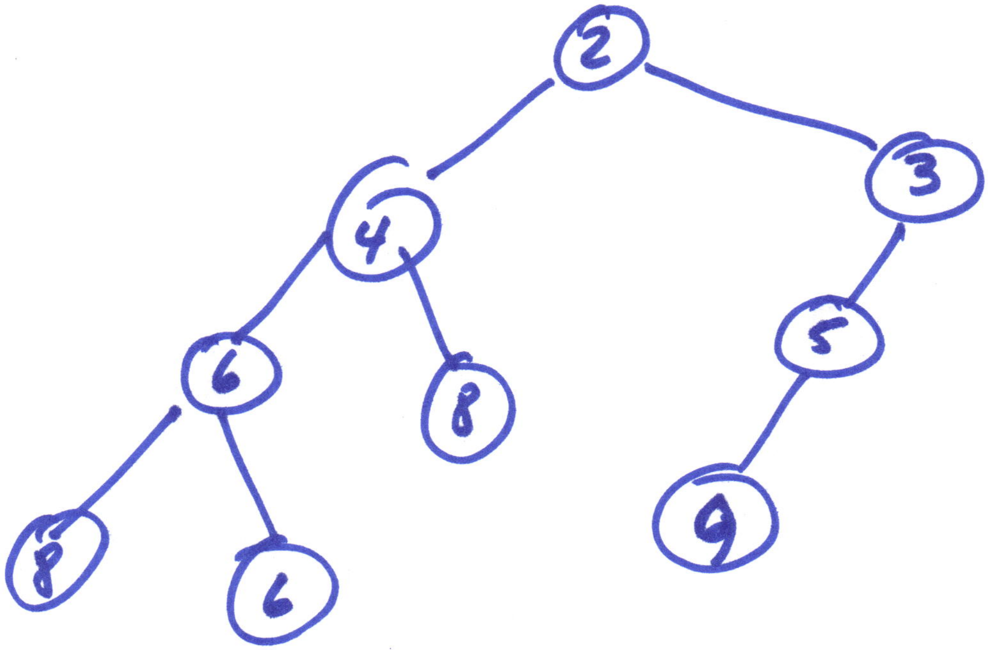
* THE RIGHTMOST PATH HAS LENGTH $O(\log n)$, n is # of internal nodes.

PROP #2 $\Rightarrow n \geq 2^{s(root)} - 1$

$\Rightarrow s(root) \leq \log(n+1)$

PROP #1 $\Rightarrow$ Rightmost path has length s(root)

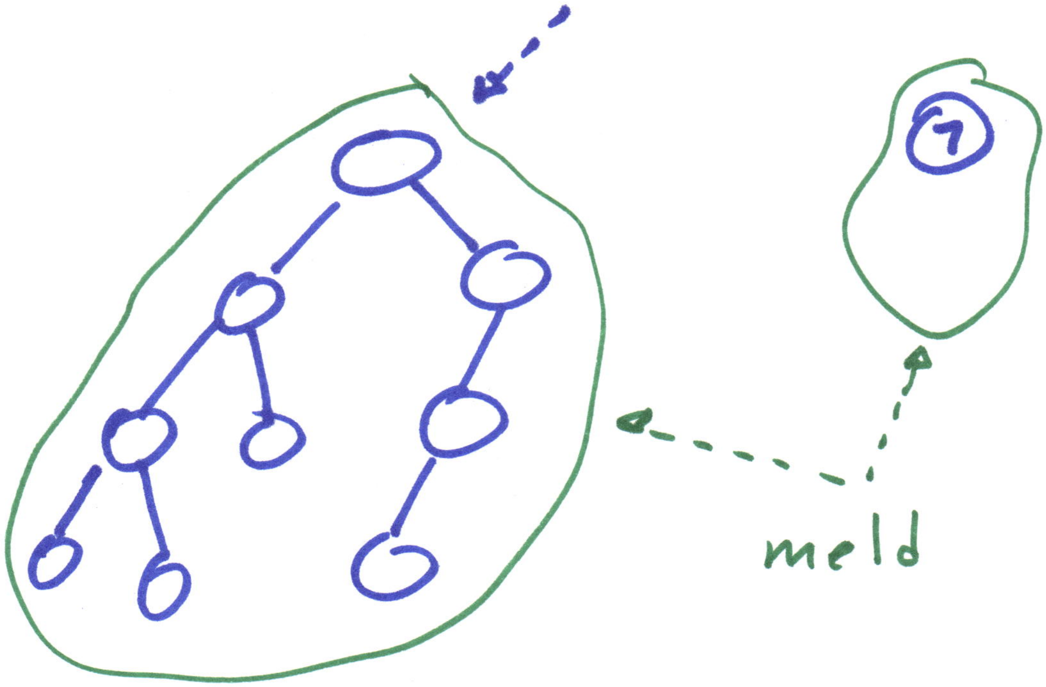
MIN HBLT



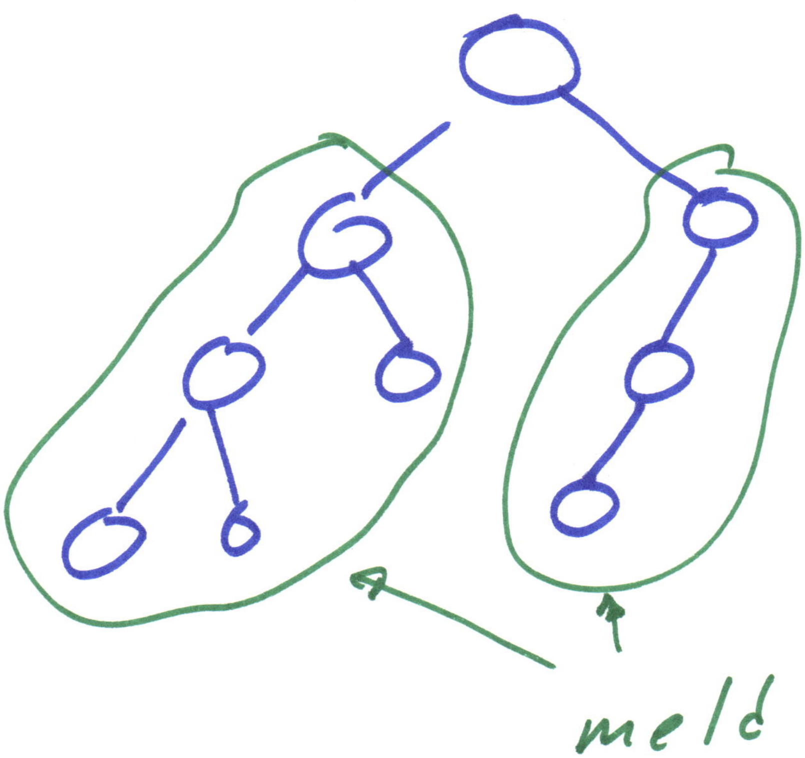
Insert
Delete Min
MELD( )

OPERATIONS

INSERT $x = 7$

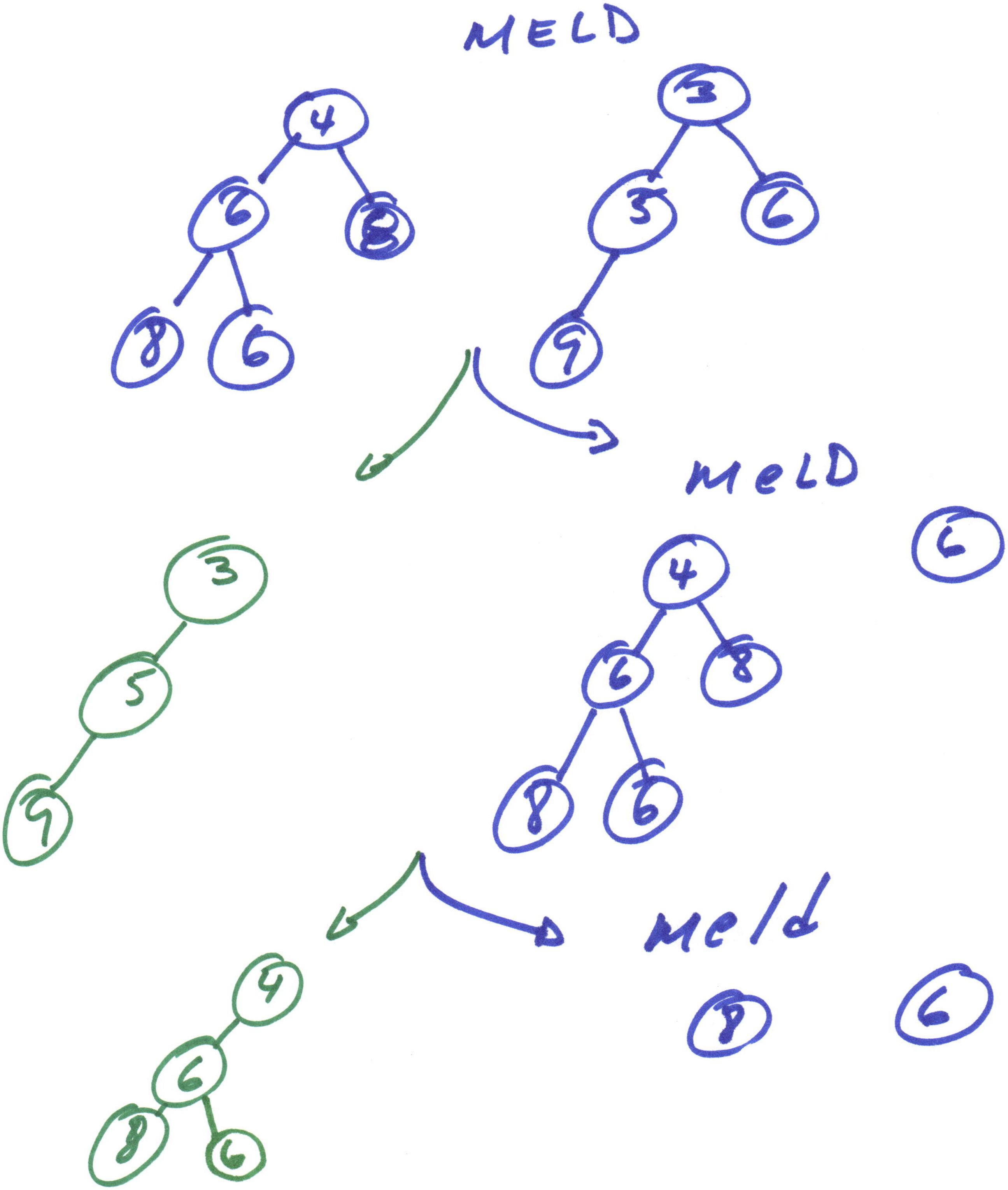


DELETE MIN

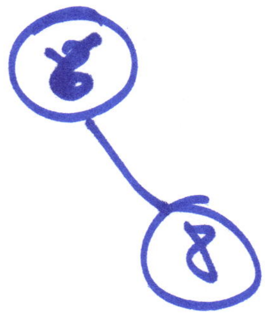


MELD TWO HBLTs

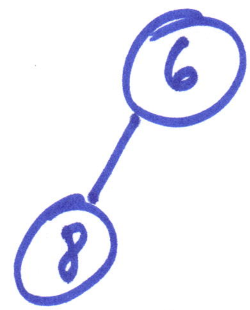
"TRAVERSE RIGHTMOST PATHS"



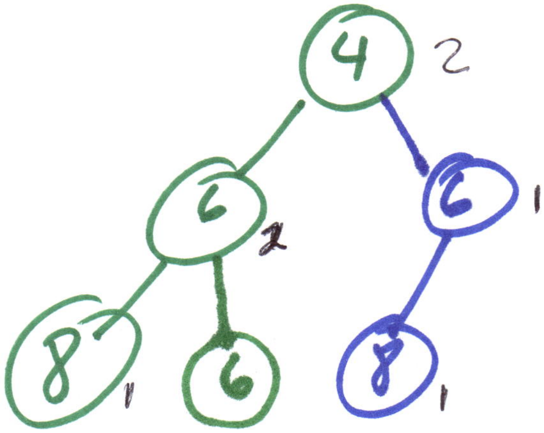
MELD



BUT NOT HBLT
SO SWAP CHILDREN

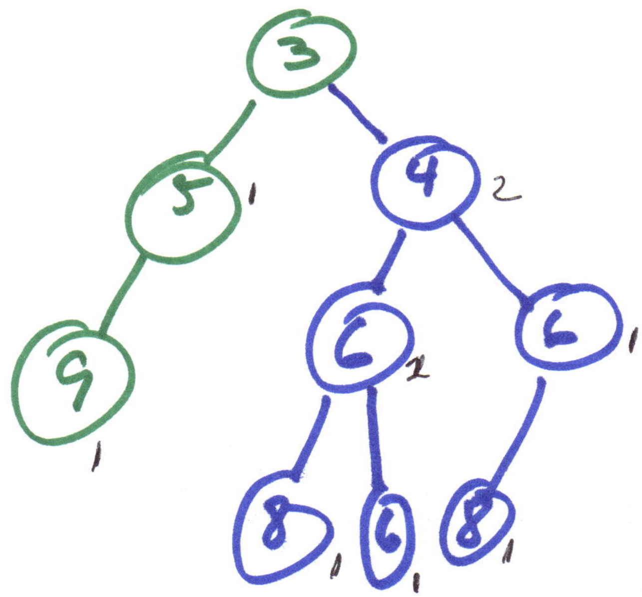


Combine with previous

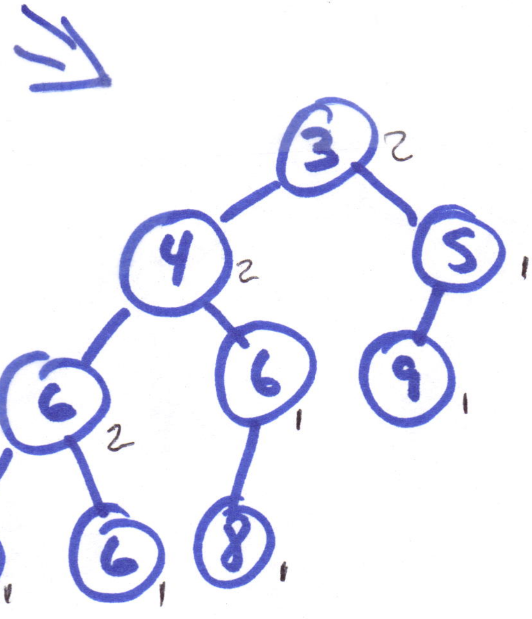


Swap not needed!

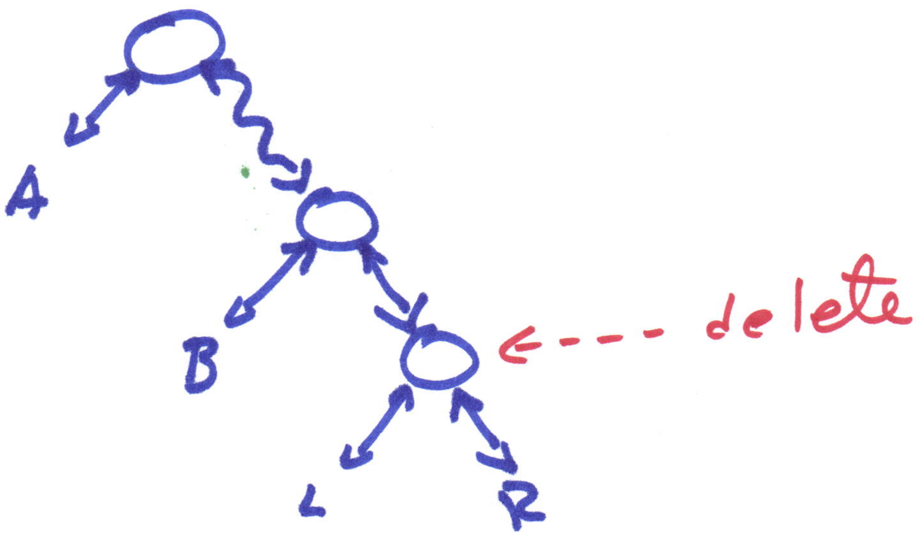
Combine with previous

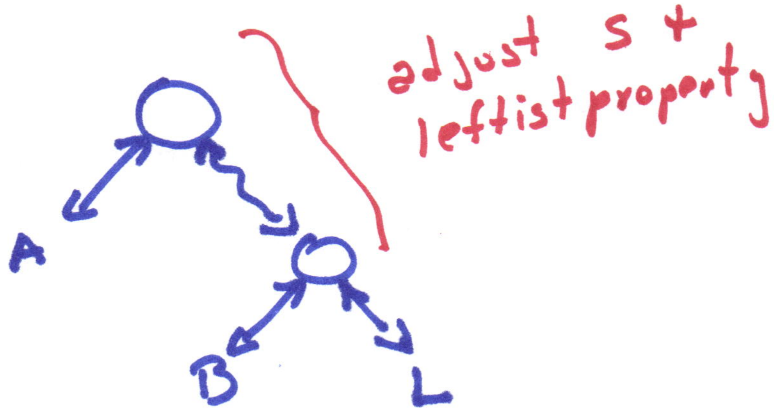


Swap needed!



ARBITRARY DELETE





⇓ now meld resulting HBLT + R