

## Red-Black Trees

Red-black trees are binary search trees that satisfy:

1. Every node is either red or black.
2. If a node is red, then its parent is black.
3. For a given node, every path to a NIL has the same number of black nodes, called black-height.

A red-black tree with  $n$  nodes has  $h \leq 2 \lg(n + 1)$ .

Proof: Suppose r-b tree with black-height  $b$ .

Point 1:  $h \leq 2b$ .

Length  $h$  path has  $\geq h/2$  black nodes.

Point 2: Black-height  $b$  implies  $n \geq 2^b - 1$

Basis: If  $b = 1$ , then  $n \geq 1 = 2^1 - 1$

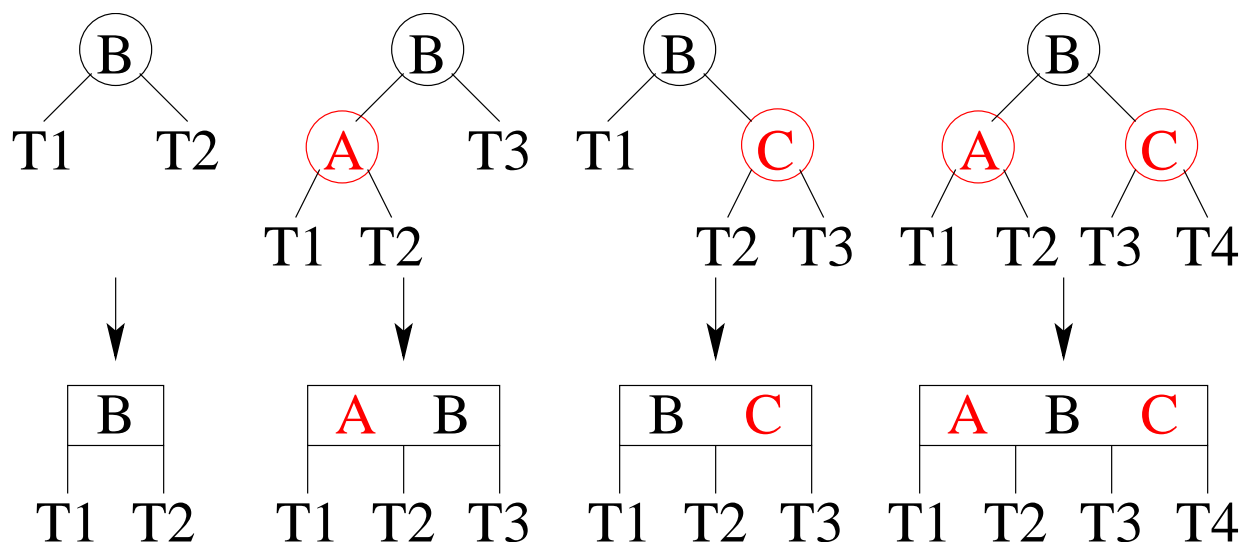
Assume: If  $b = k - 1$ , then  $n \geq 2^{k-1} - 1$

Show: If  $b = k$ , then  $n \geq 2^k - 1$

Induction: 2 subtrees with black-height  $k - 1$   
 implies  $n \geq 1 + 2(2^{k-1} - 1) = 2^k - 1$

$n \geq 2^b - 1 \geq 2^{h/2} - 1$  implies  $h \leq 2 \lg(n + 1)$

To explain the algorithms, define a *cluster* to be a black node and its red children.



RB-INSERT( $T, x$ )

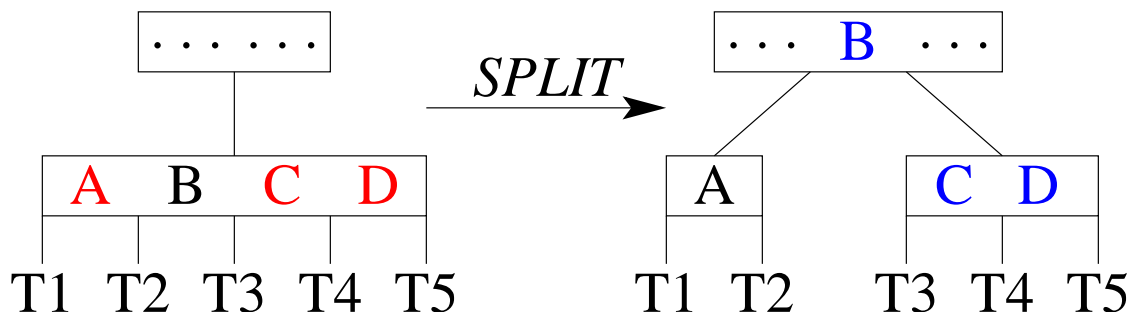
  TREE-INSERT( $T, x$ ) into a “leaf” cluster

*cluster*  $\leftarrow$  cluster containing  $x$

**while** *cluster* has too many nodes

**do** “split” *cluster*

*cluster*  $\leftarrow$  parent of *cluster*



RB-DELETE( $T, z$ )

  TREE-DELETE( $T, z$ )

$cluster \leftarrow$  leaf cluster with one less node

**while**  $cluster$  is empty

**do** “splice” or “shift” into  $cluster$

$cluster \leftarrow$  parent of  $cluster$

