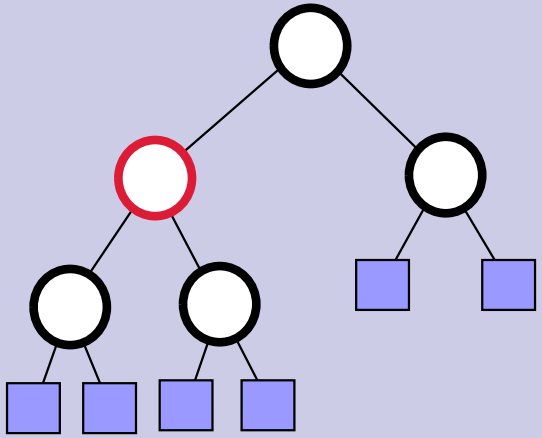
Red-Black Trees

- What are they?
- Their relation to 2-4 trees
- Deletion in red-black trees

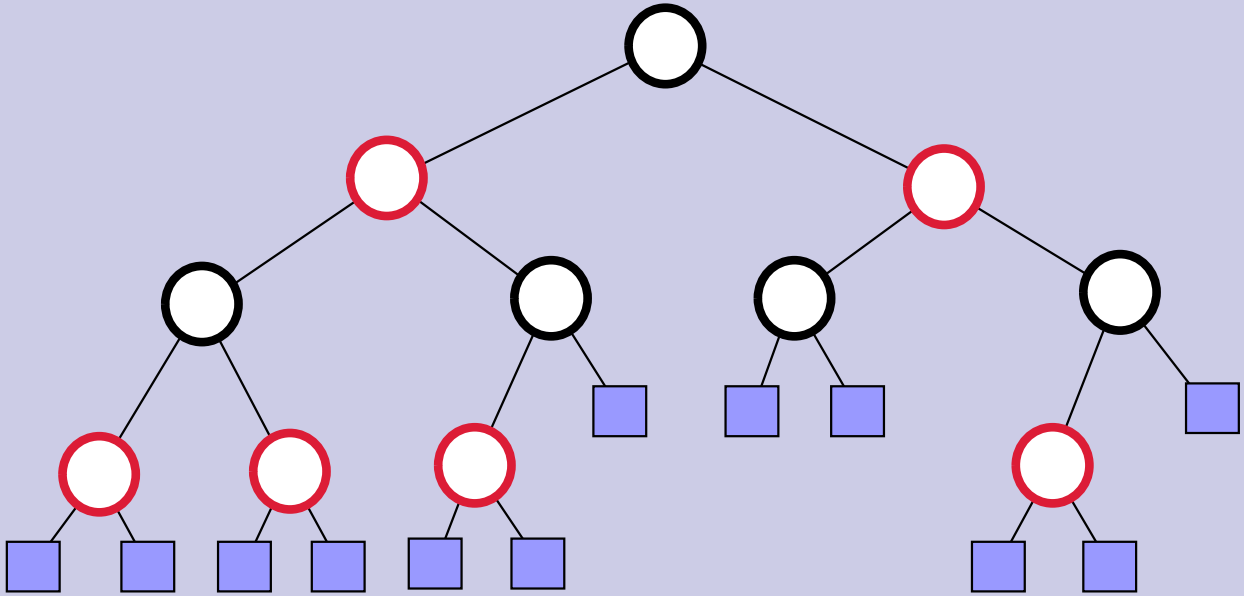
Red-Black Trees

- A red black tree is a binary search tree in which each node is colored red or black.
- The root is colored black.
- A red node can have only black children.
- If a node of the BST does not have a left and/or right child then we add an external node.
- External nodes are not colored.
- The black depth of an external node is defined as the number of black ancestors it has.
- In a red-black tree every external node has the same black depth.

Examples of red-black trees

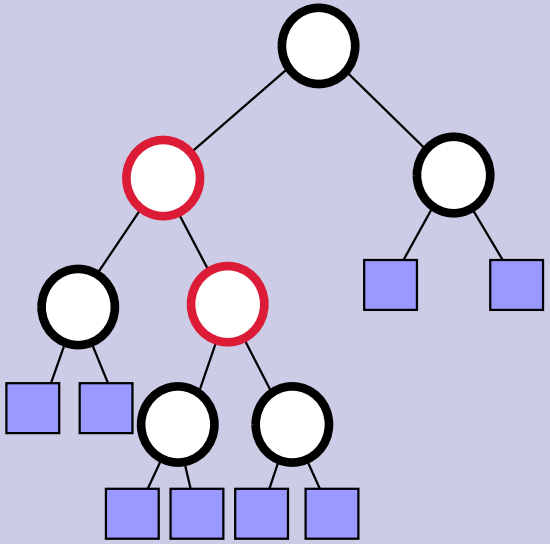


Black height of tree is 2

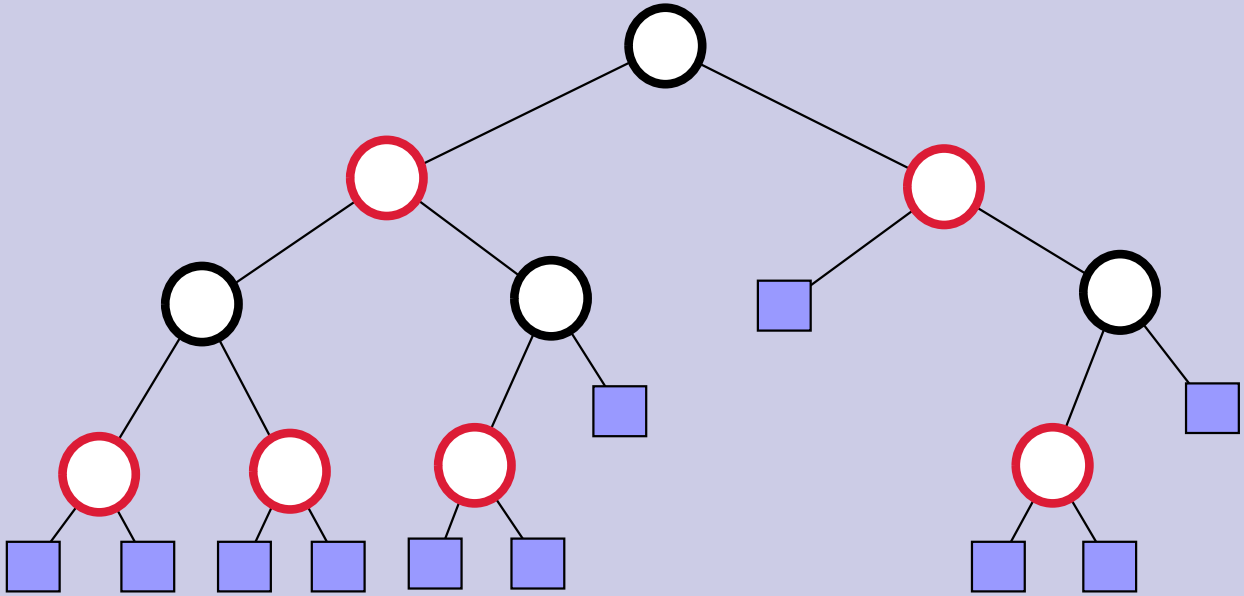


Black height of tree is 2

Trees which are not red-black



Double red

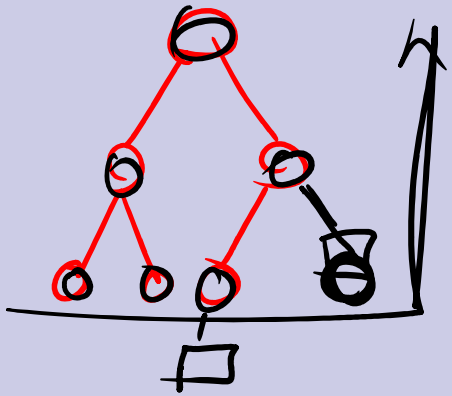


Black height not uniform

Height of a red-black tree

- Let h be the black height of a red-black tree on n nodes.
- n is smallest when all nodes are black. In this case tree is a complete binary tree of height h and $n=2^h -1$.
- n is largest when alternate levels of tree are red. Then height of tree is 2h and $n=2^{2h}-1$.
- Hence, $\log_4 n < h < 1+\log_2 n$

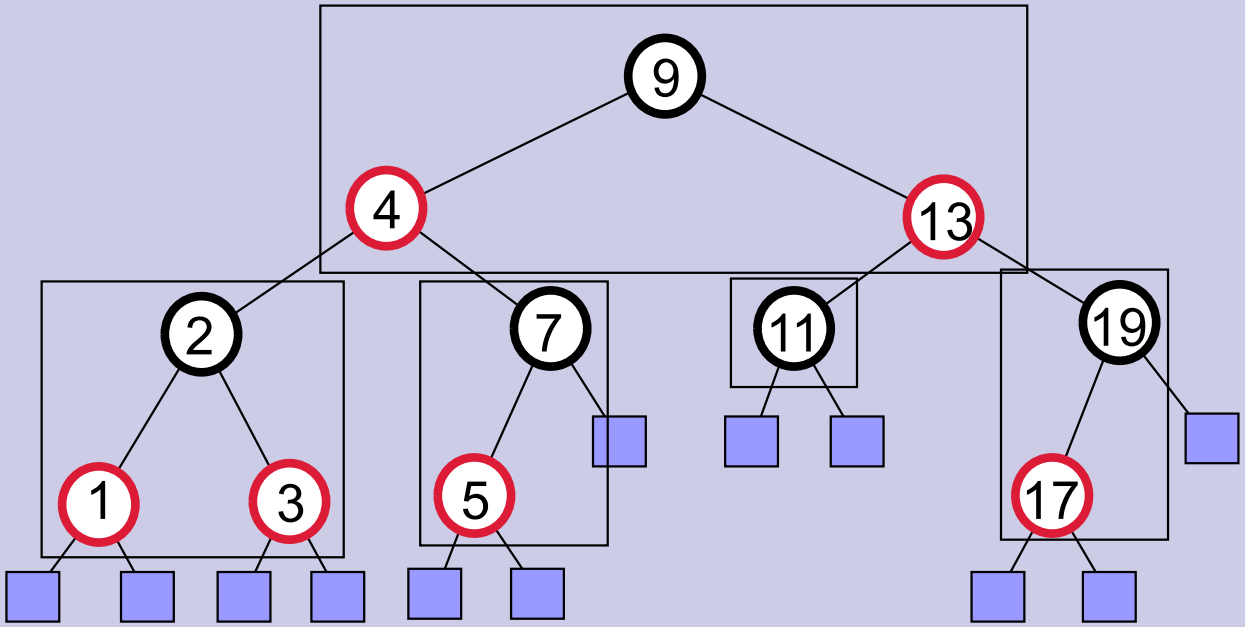
$$2^{h-1} < n < 2^{2h} - 1$$

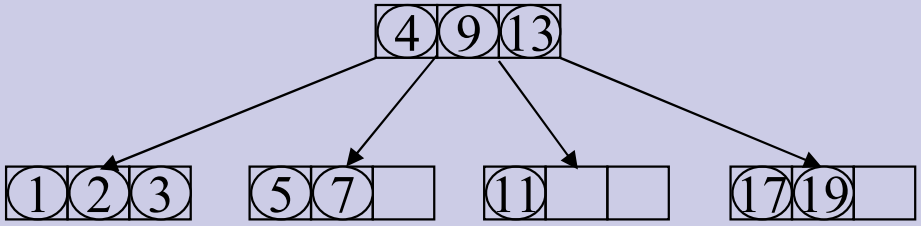


Red-black trees to 2-4 trees

- Any red-black tree can be converted into a 2-4 tree.
- Take a black node and its red children (at most 2) and combine them into one node of a 2-4 tree.
- Each node so formed has at least 1 and at most 3 keys.
- Since black height of all external nodes is same, in the resulting 2-4 tree all leaves are at same level.

Example

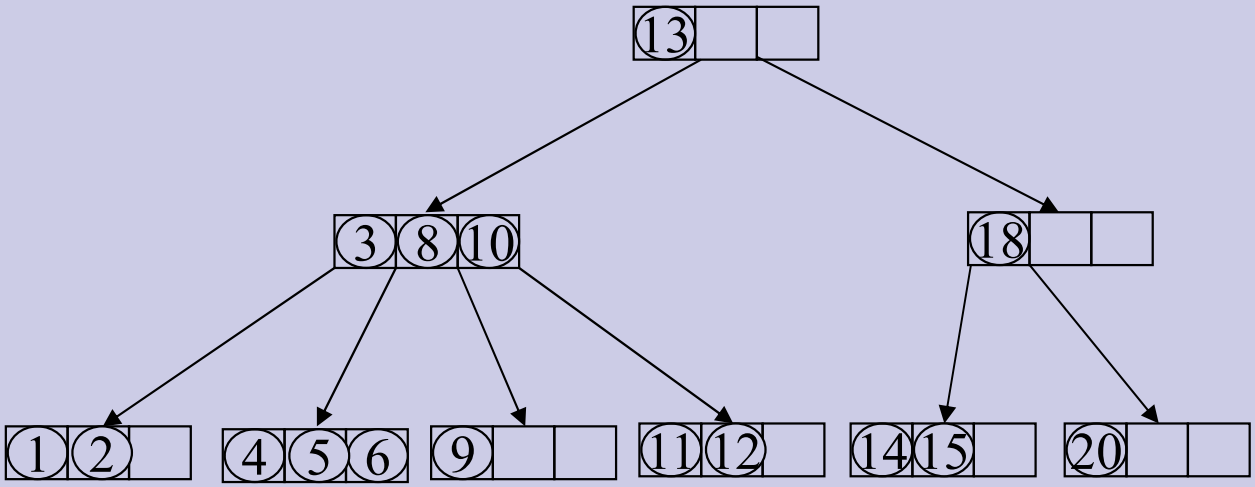


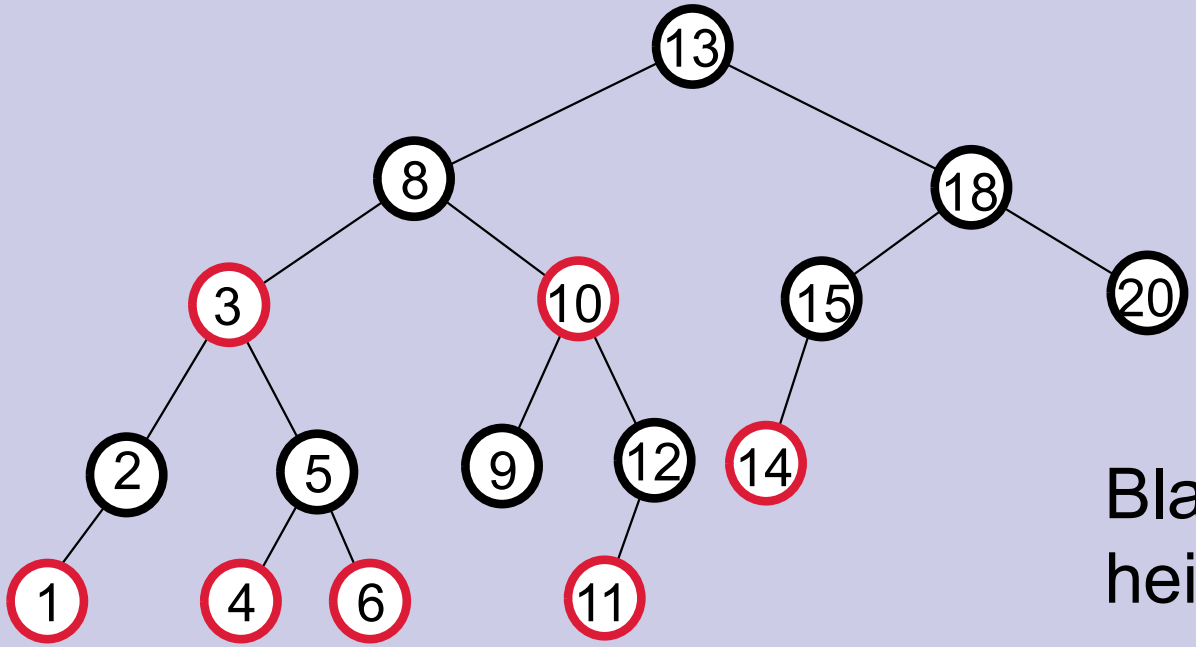


2-4 trees to red-black trees

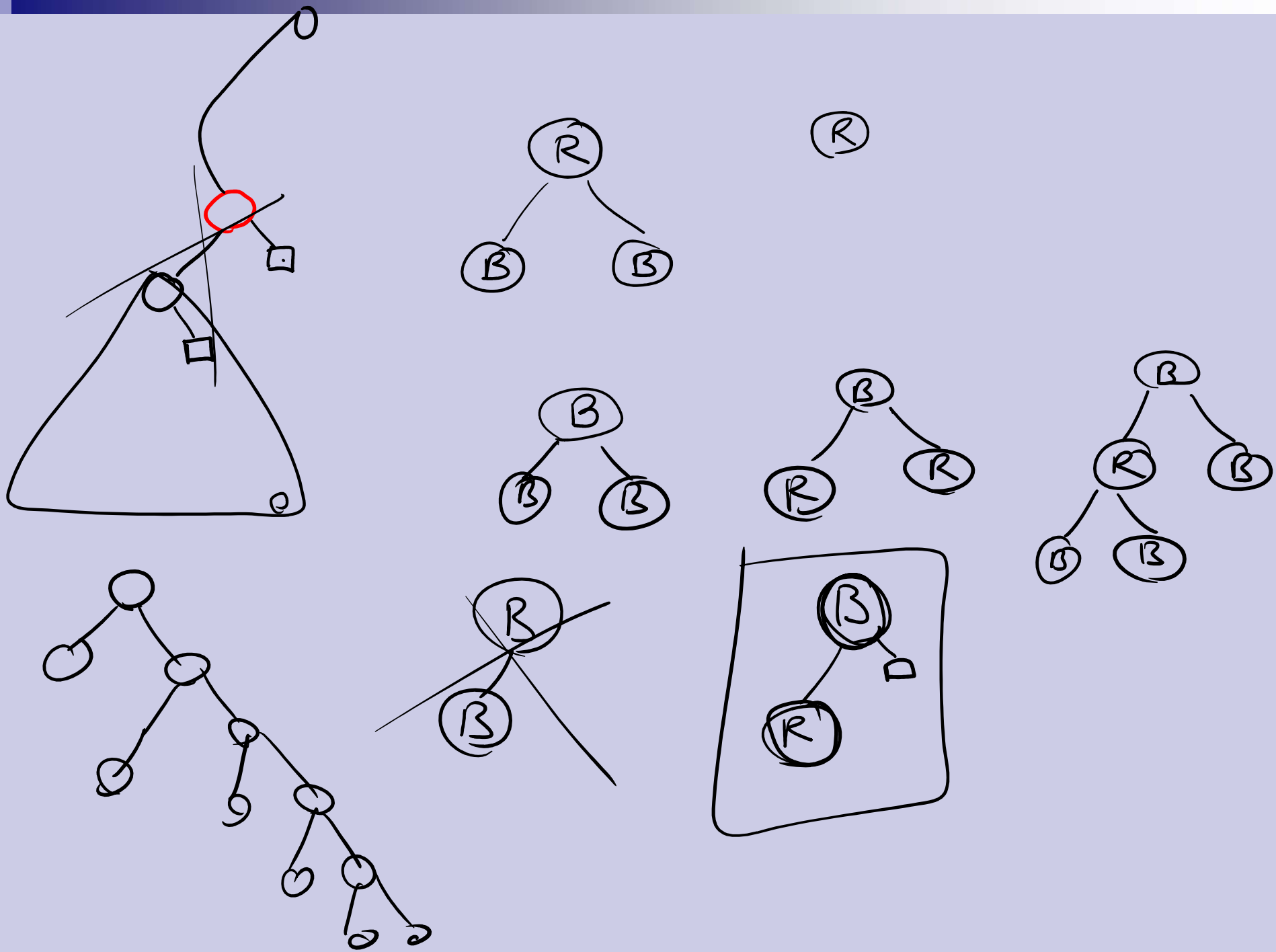
- Any 2-4 tree can be converted into a red-black tree.
- We replace a node of the 2-4 tree with one black node and 0/1/2 red nodes which are children of the black node.
- The height of the 2-4 tree is the black height of the red-black tree created.
- Every red node has a black child.

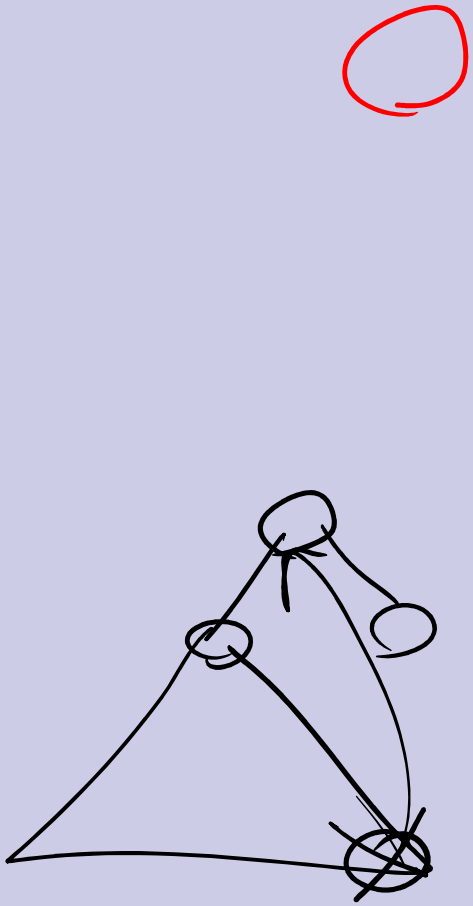
Example

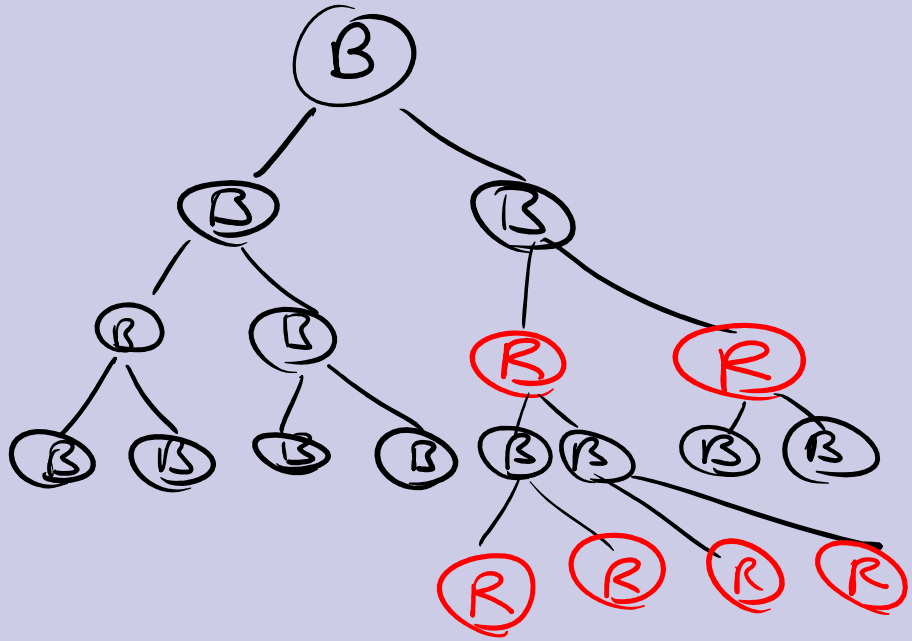


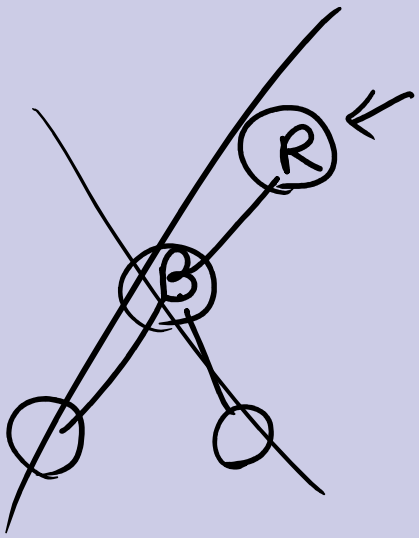


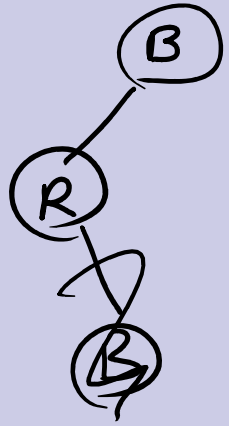
Black height is same as height of 2-4 tree


R
B
B
R
B
B
B
B
R
R
B
R
B
B
B
R
B
B
R



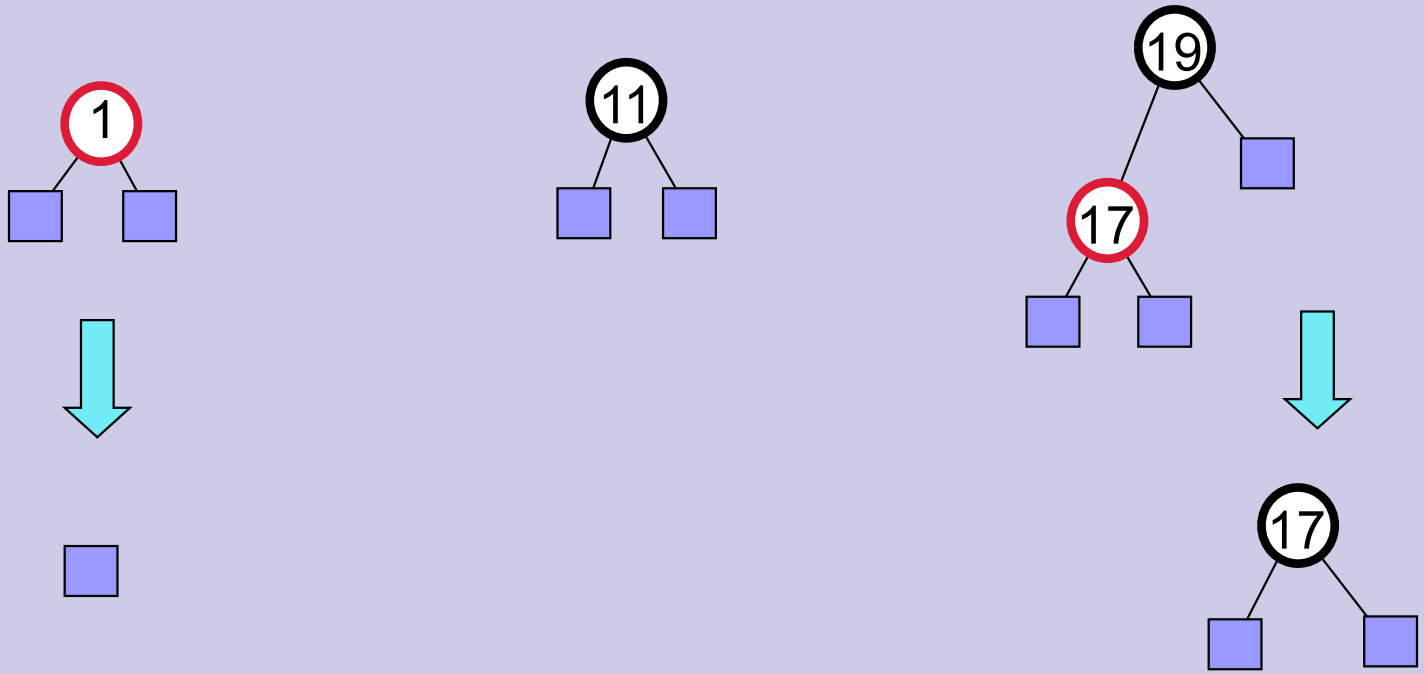

B
B
B
R
B
R
R
B
B
B
B
B
B
B
R
R
R
R


R
B


B
R
B

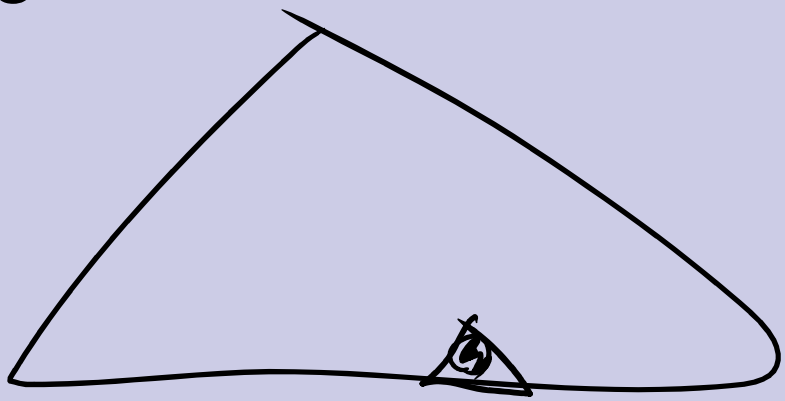
Deletion: preliminaries

- To delete a node we proceed as in a BST.
- Thus the node which is deleted is the parent of an external node.
- Hence it is either a leaf or the parent of a leaf.

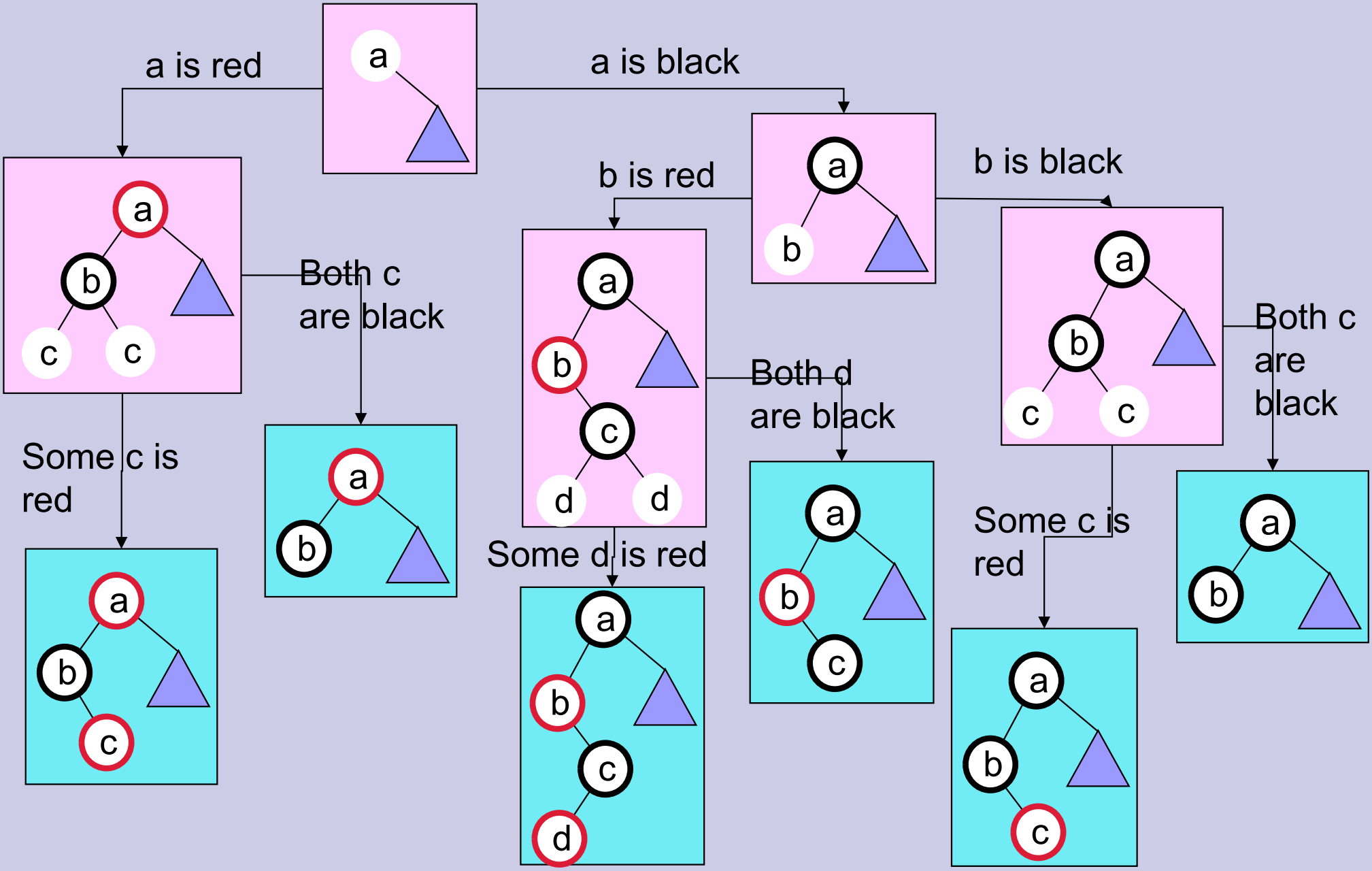


Deletion: preliminaries(2)

- Hence we can assume that the node deleted is a black leaf.
- Removing this reduces black depth of an external node by 1.
- Hence, in a general step, we consider how to reorganize the tree when the black height of some subtree goes down from h to h-1.

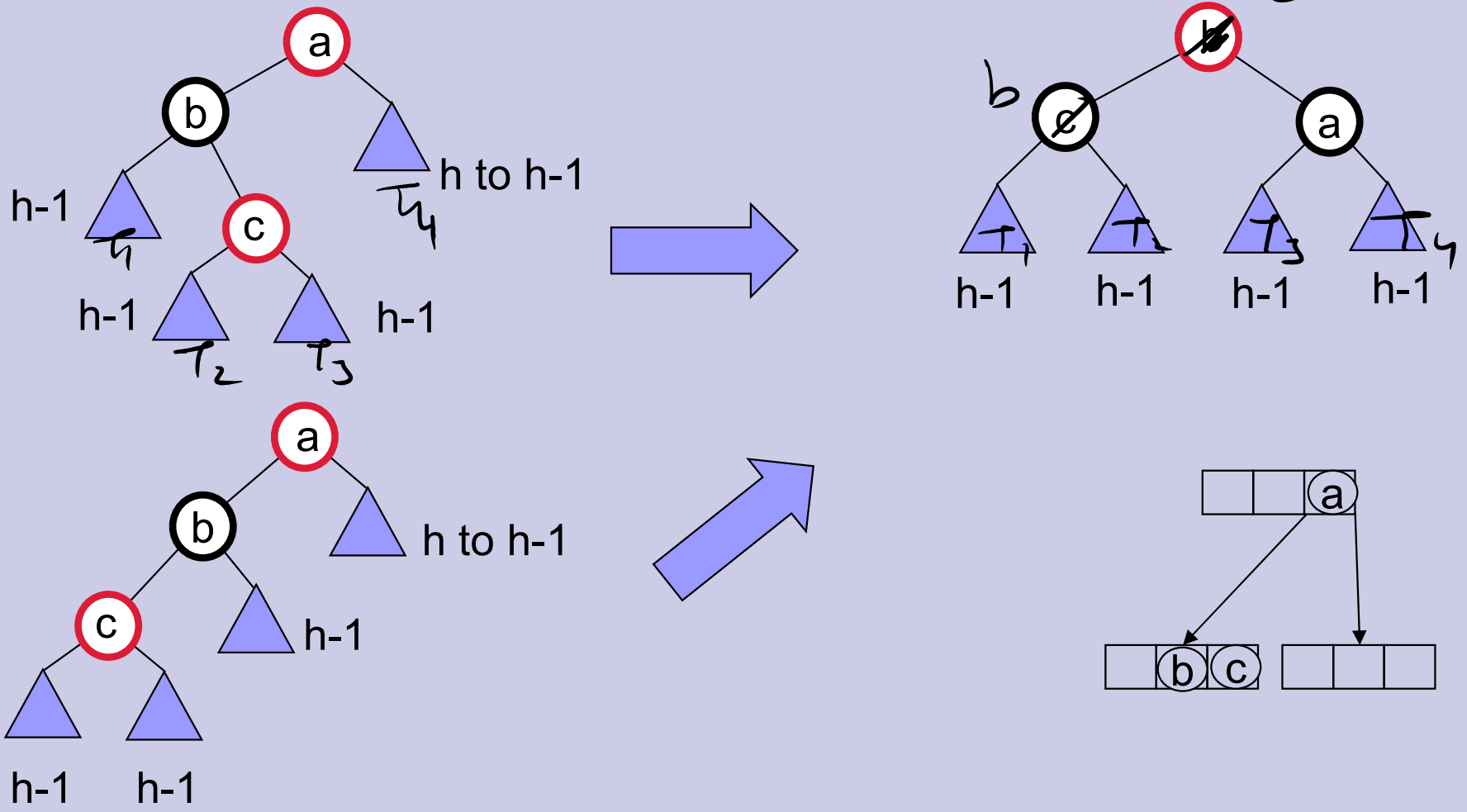


Deletion: the cases



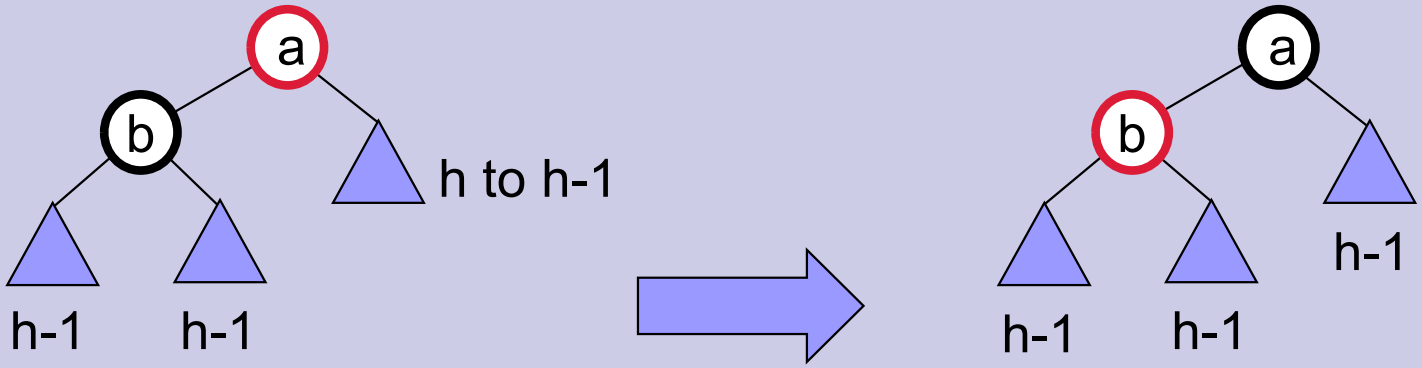
Deletion: case1.1

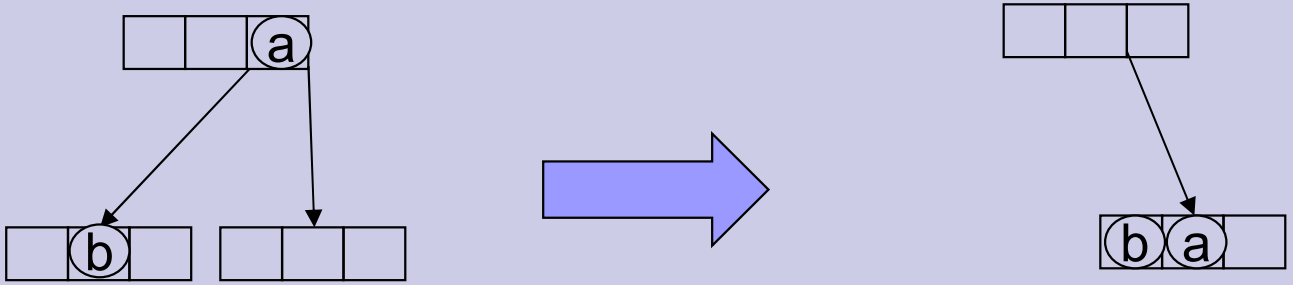
- If parent is a red node (a).
- Then it has a child (b) which must be black
- If b has a red child (c)



Deletion: case 1.2

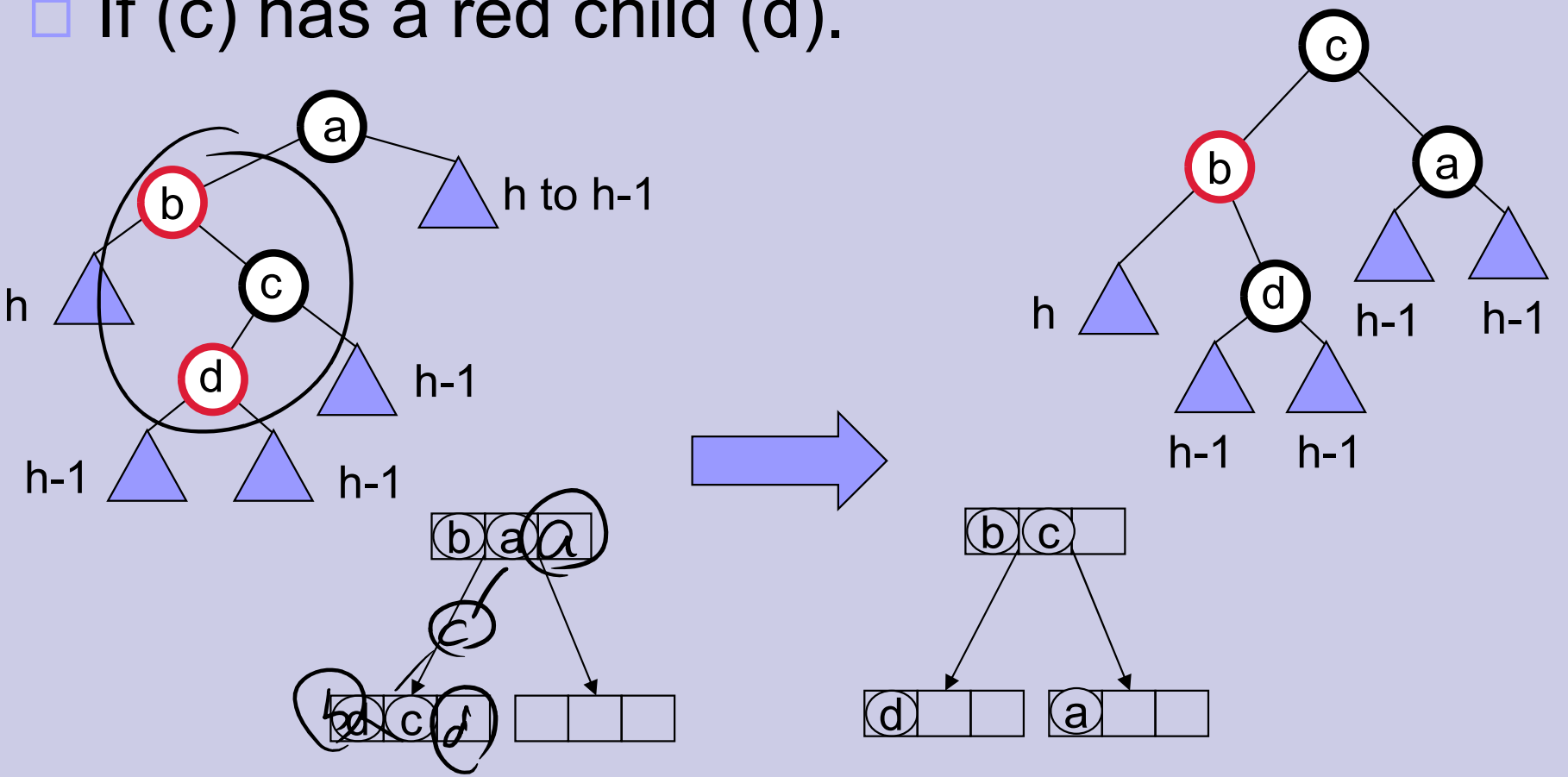
- If parent is a red node (a).
- Then it has a child (b) which must be black
- If b has no red child





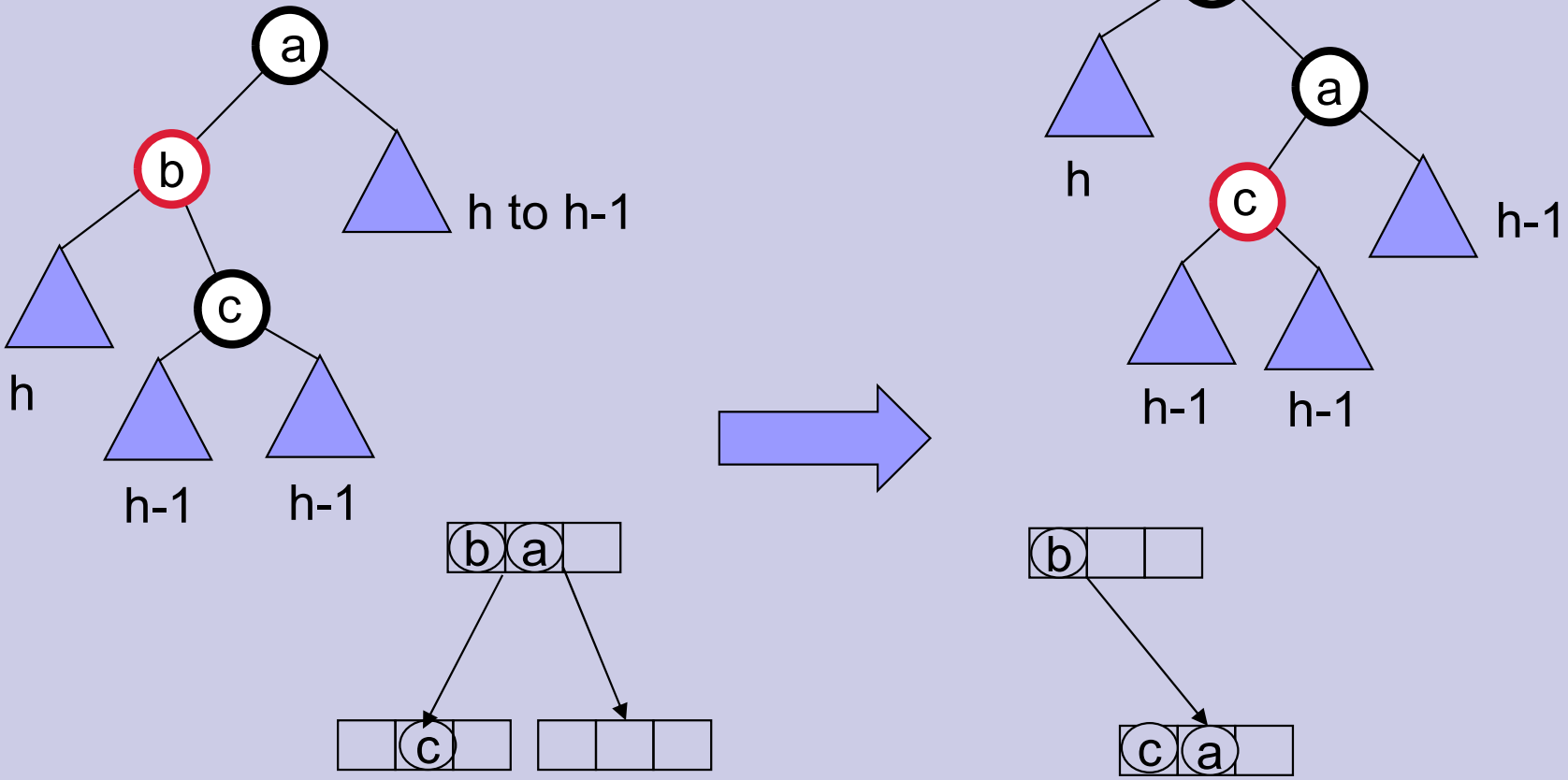
Deletion: case 2.1.1

- If parent is a black node (a).
- If a has a red child (b)
- Consider right child of b (c) which must be black.
- If (c) has a red child (d).



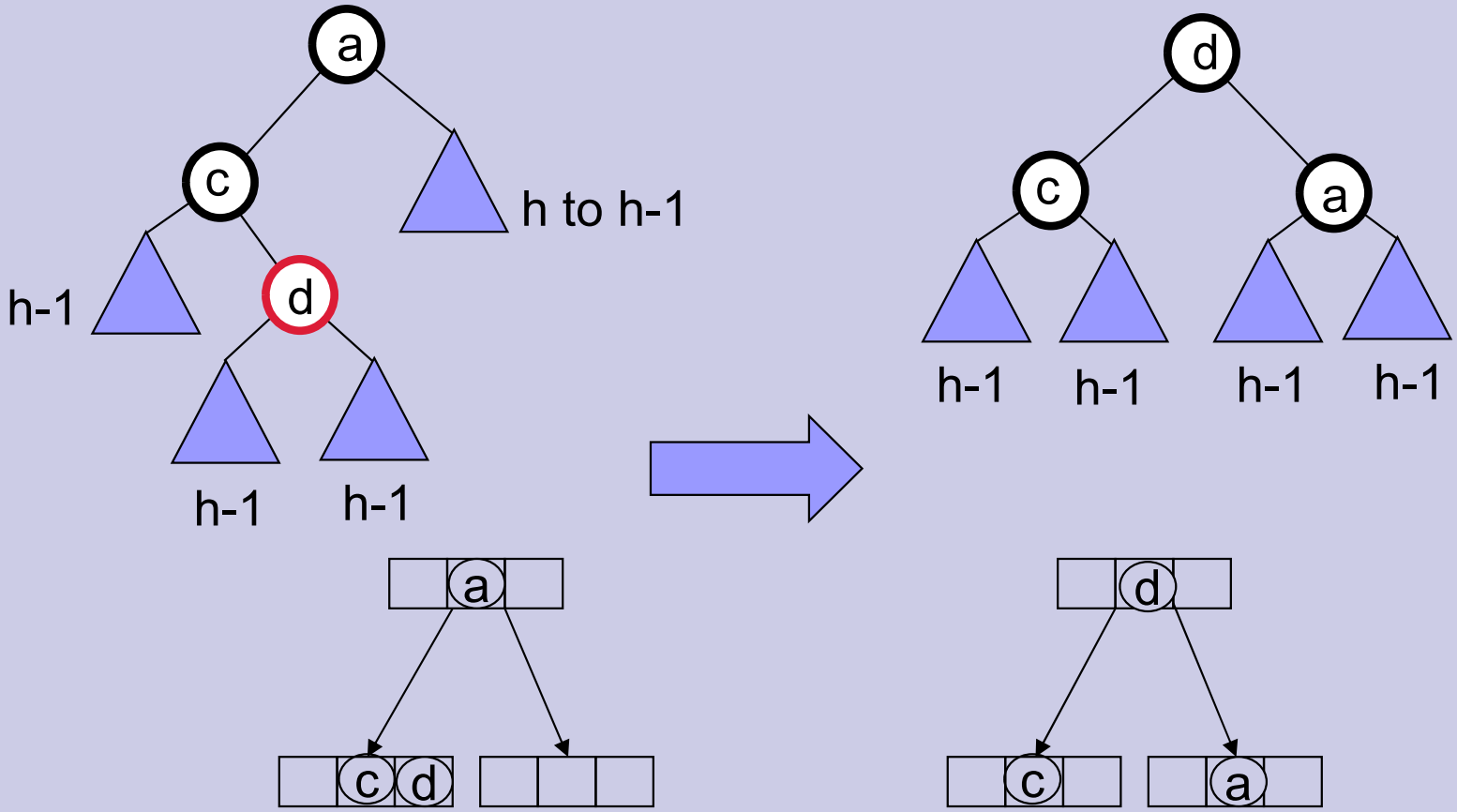
Deletion: case 2.1.2

- If parent is a black node (a).
- If a has a red child (b)
- Consider right child of b (c) which must be black.
- If (c) has no red child.



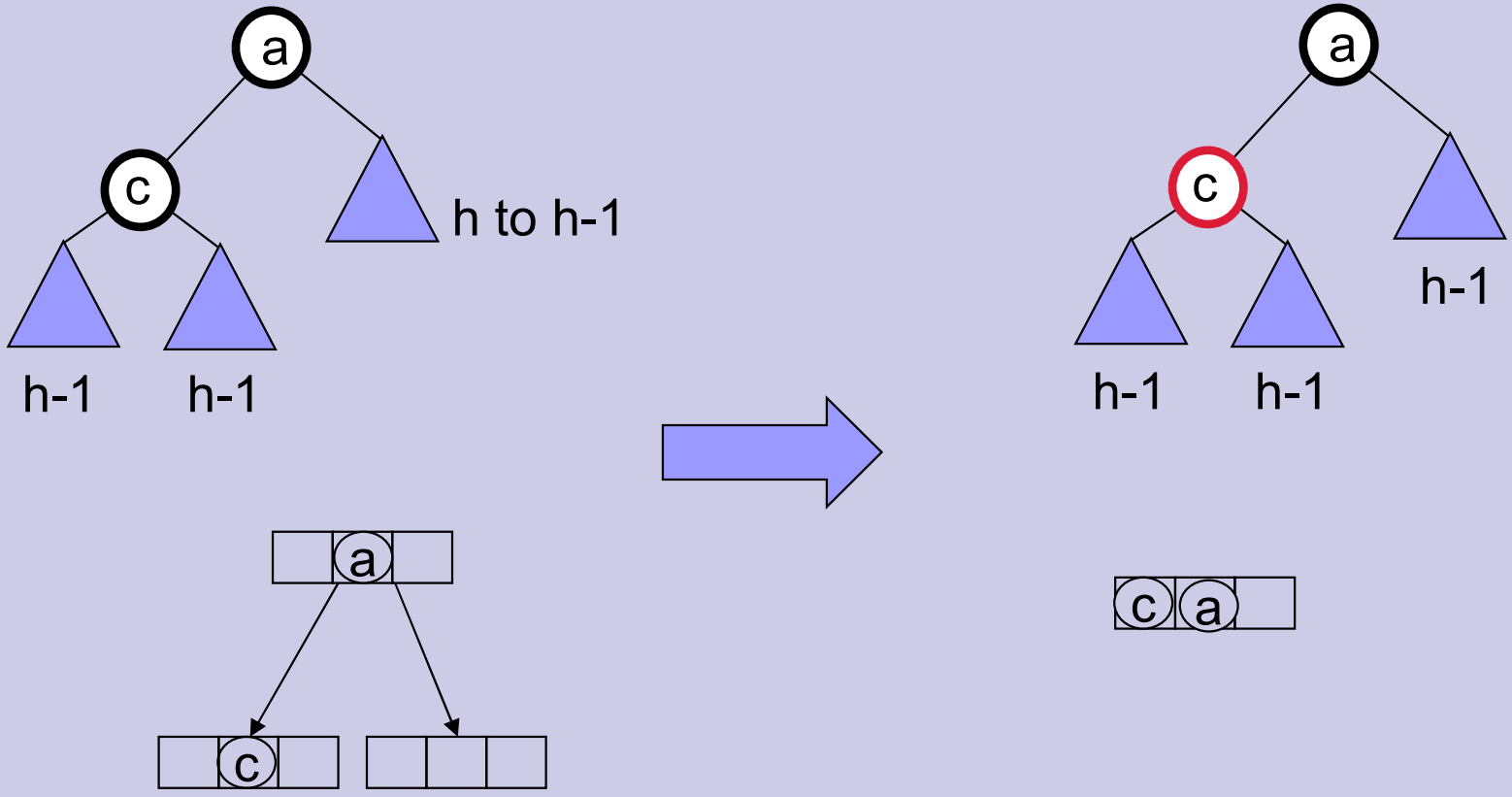
Deletion: case 2.2.1

- If parent is a black node (a).
- If the child of node a (c) is black.
- If (c) has a red child (d)



Deletion: case 2.2.2

- If parent is a black node (a).
- If the child of node a (c) is black.
- If (c) has no red child.



Deletion: Summary

- In all cases, except 2.2.2, deletion can be completed by a simple rotation/recoloring
- In case 2.2.2, the height of the subtree reduces and so we need to proceed up the tree.
- But in case 2.2.2 we only recolor nodes.
- Thus, if we proceed up the tree then we only need to recolor. Eventually we would do a rotation.