COL106
Data Structures and Algorithms

Subodh Sharma and Rahul Garg

Announcements

Announcements

- Lab examination postponed to after the mid-semester examination due to logistical issues
- Minor syllabus
  - Everything covered till today, plus
  - 2-4 Trees, a-b Trees and Red-Black (till today)

Red-Black Trees

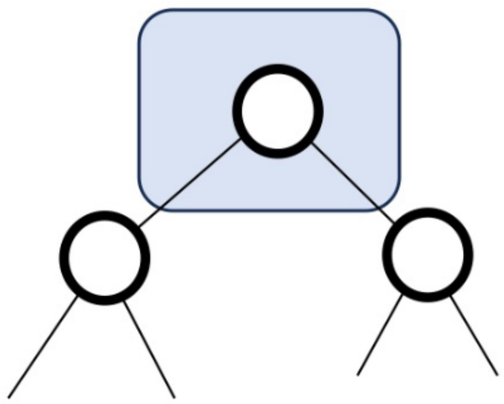
Red-Black Trees

- Binary Search trees
- Each node is either colored red or black
- The root is colored black
- A red node can have only black children
- A black node can have red or black children

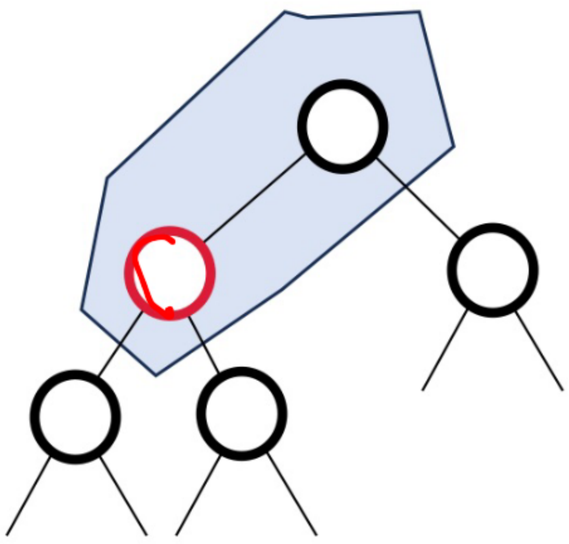
Red-Black Trees

- External nodes
  - Added as left child of nodes that do not have a left child
  - Added as right child of nodes that do not have right child
  - Leaves will have two external nodes as children
  - Fictitious nodes only for analysis
  - Do not store any keys
- External nodes are not colored
- Three types of nodes: Internal node, leaf, external nodes
- Black depth of a node: Number of black ancestors it has
- Each external node has the same “black-depth”

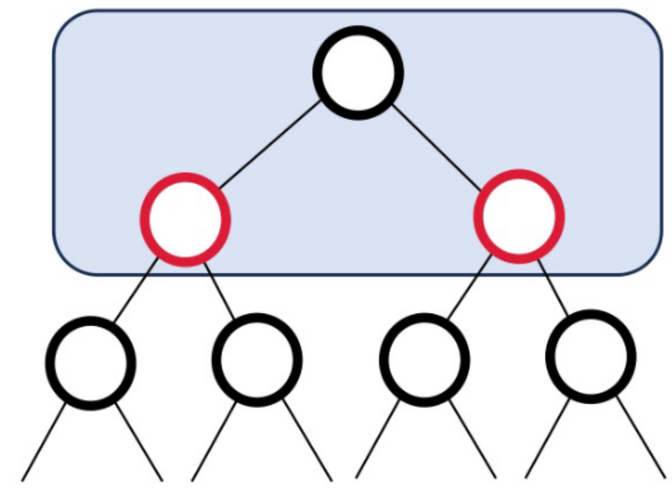
Red Black Trees



Two black descendants

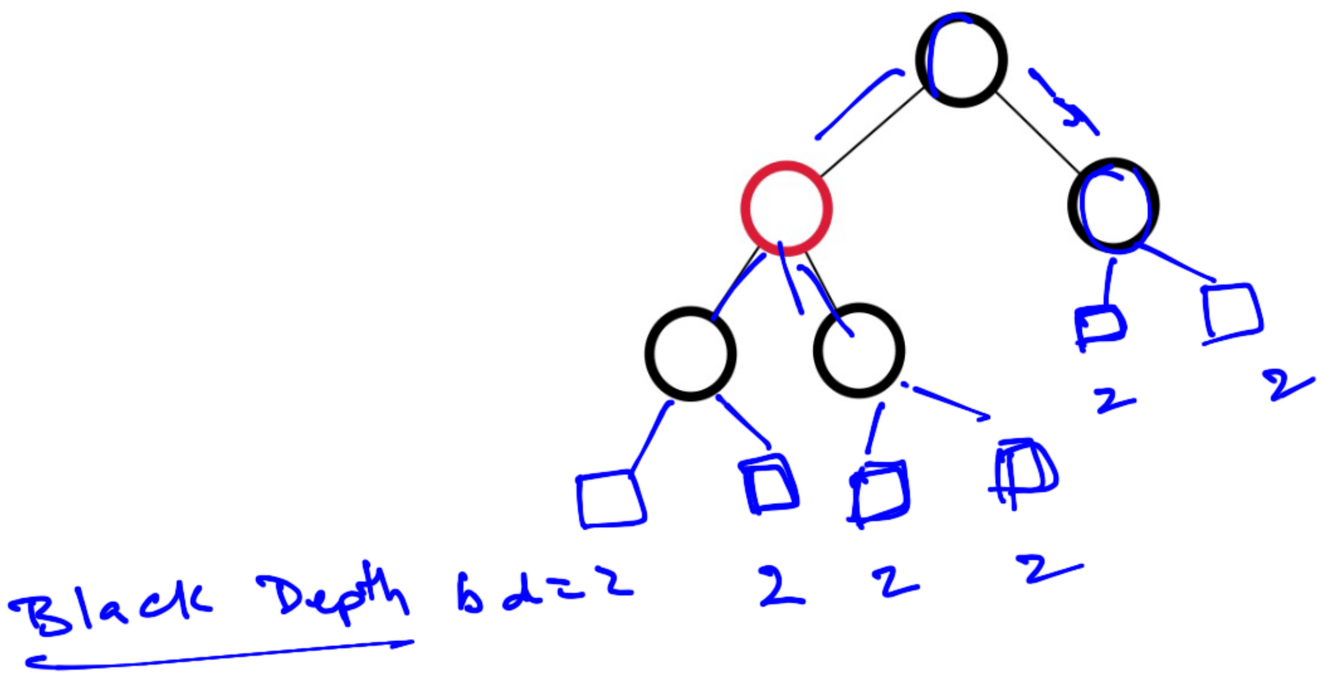


Three black descendants

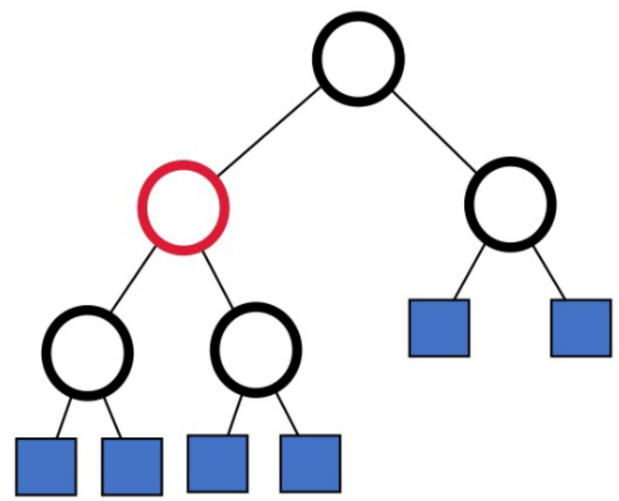


Four black descendants

Examples of Red-Black Trees

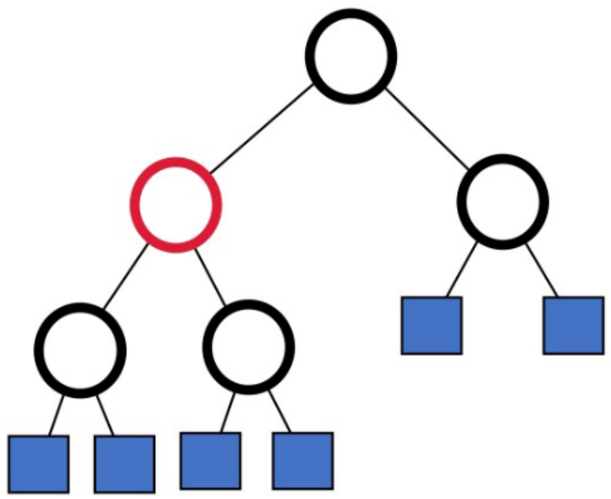


Examples of Red-Black Trees

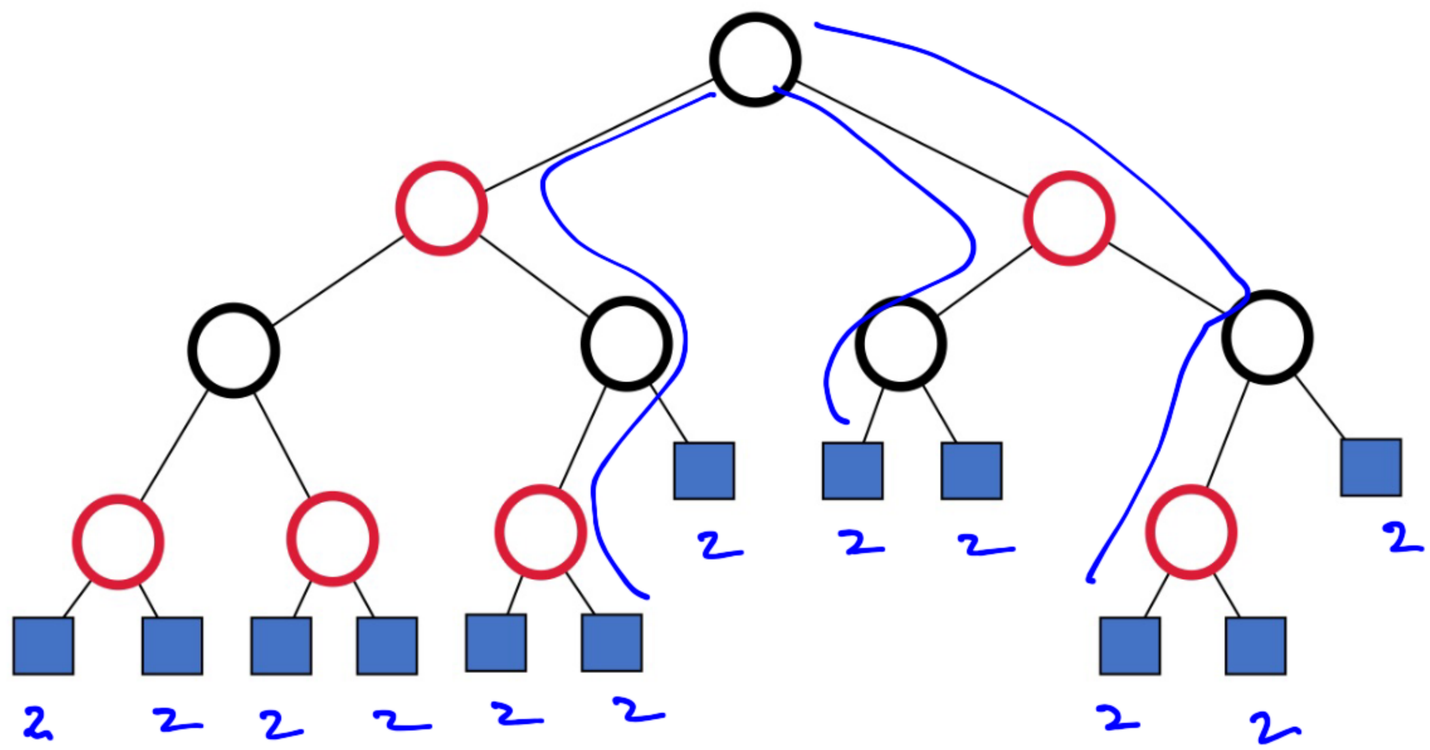


Black height of each external node is 2

Examples of Red-Black Trees



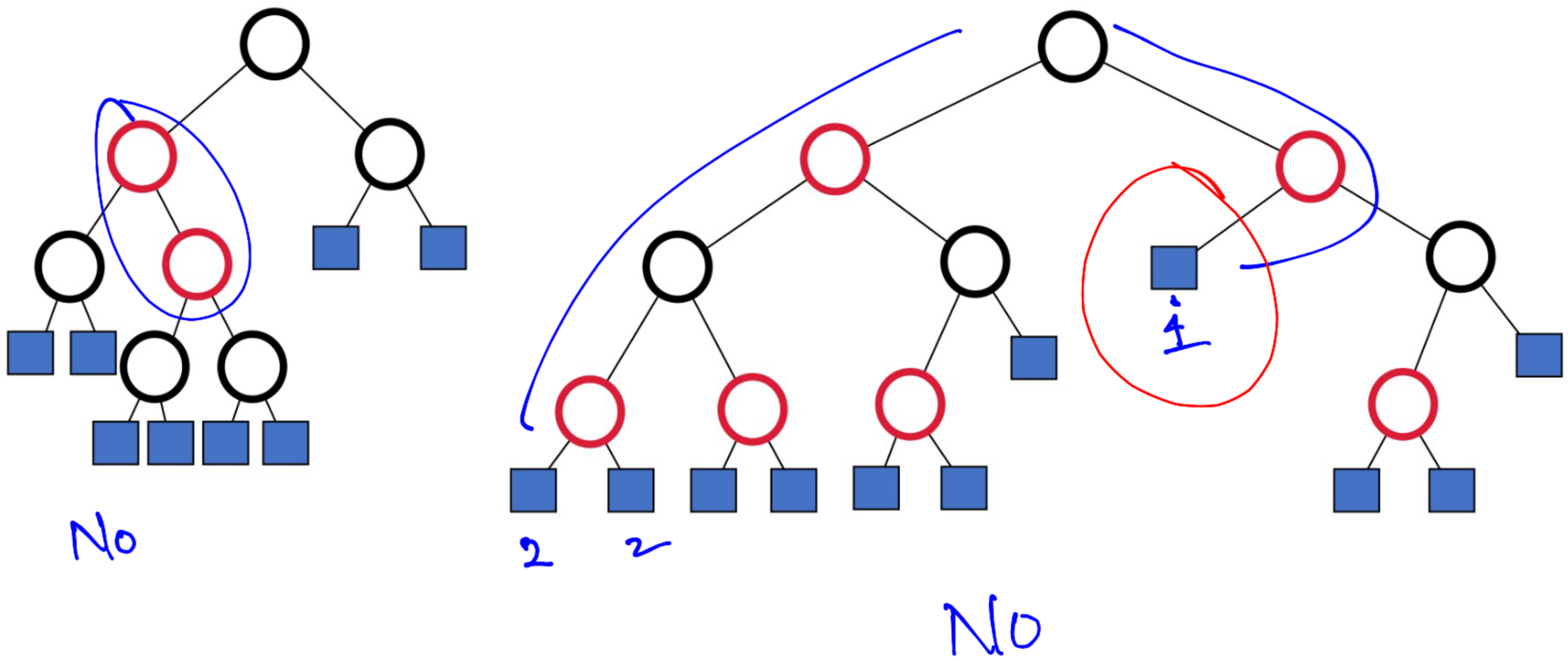
Black height of tree is 2



Black height of tree is 2

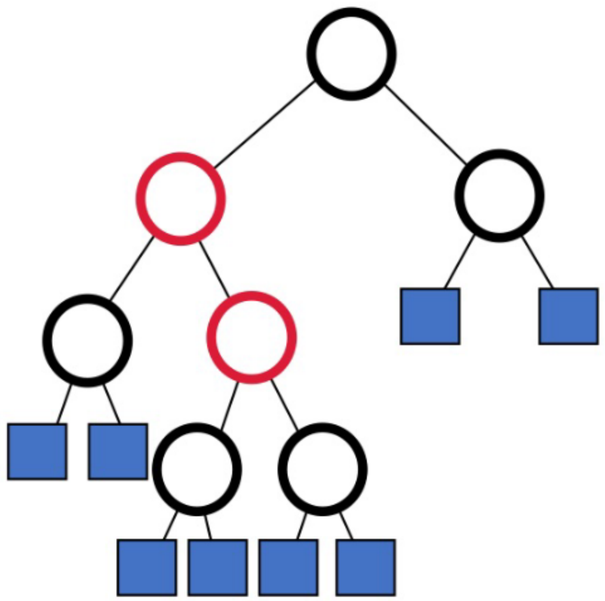
Slide adapted from Naveen Garg, CSE, IIT Delhi

Are the Following Red-Black Trees?

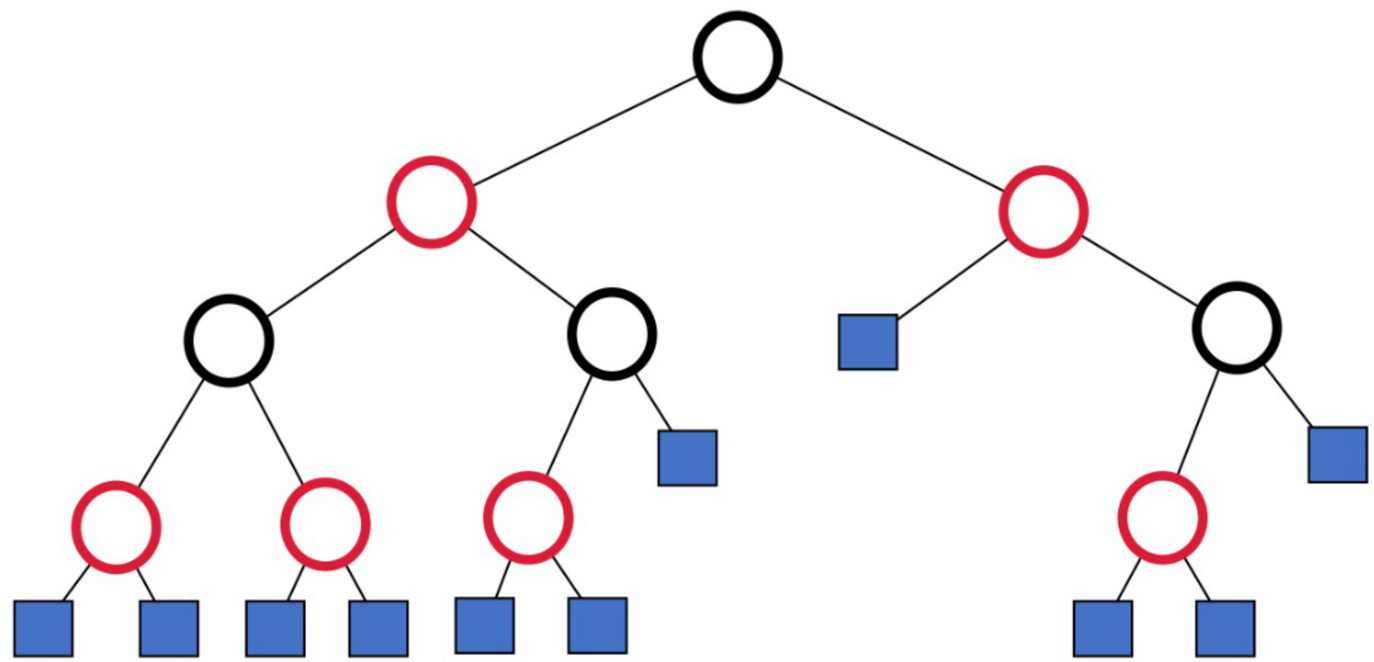


Slide adapted from Naveen Garg, CSE, IIT Delhi

Trees Which are not Red-Black



Double red



Black height not uniform

Slide adapted from Naveen Garg, CSE, IIT Delhi

Bounding the Height of a Red-Black Tree

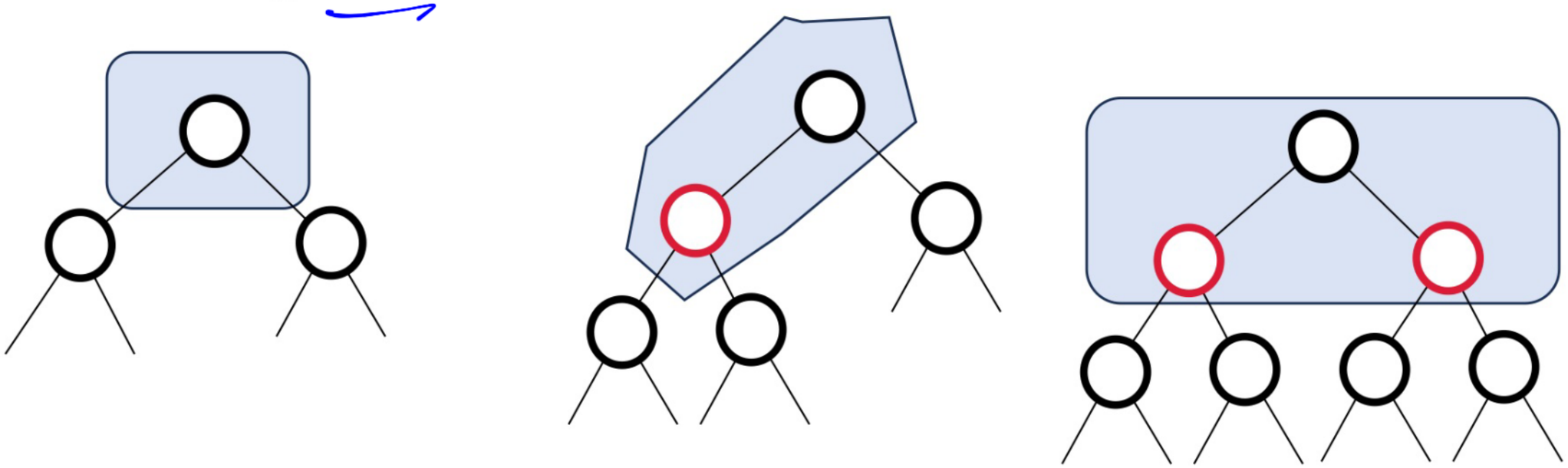
- Let h be the black height of a RB tree on n nodes
- n is smallest when all nodes are black
  - In this case tree is a complete binary tree of height h and $n=2^h-1$
- n is largest when alternate levels of tree are red
  - Then height of tree is 2h and $n=2^{2h}-1$.
- Hence, $\log_4 (n+1) \leq h \leq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

1/2

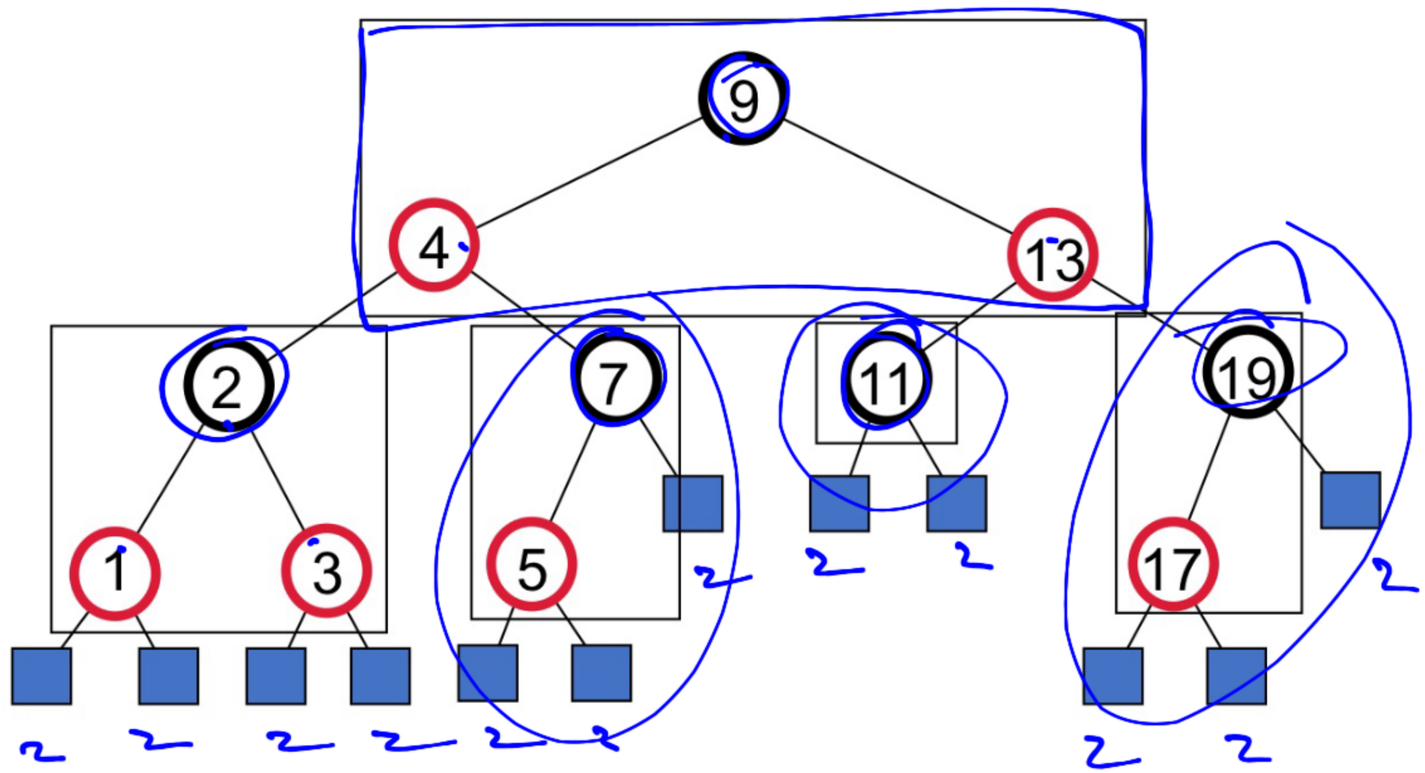
Slide adapted from Naveen Garg, CSE, IIT Delhi

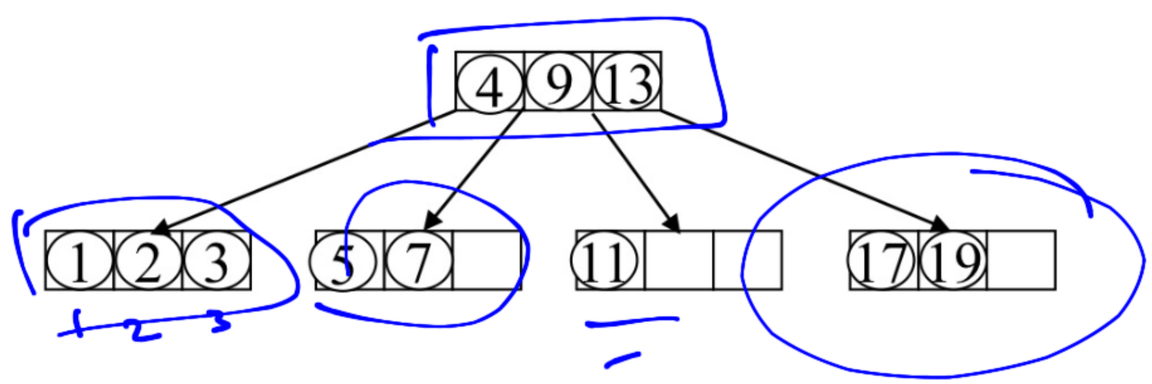
Red-Black Trees to 2-4 trees

- 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: Red-Black to 2-4 Trees





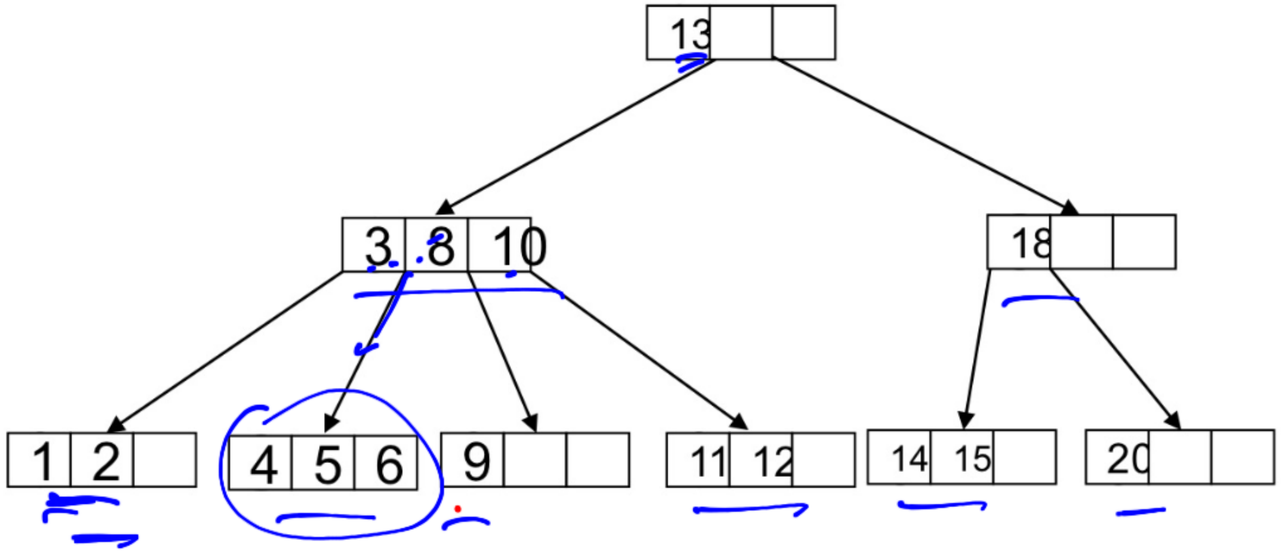
Slide adapted from Naveen Garg, CSE, IIT Delhi

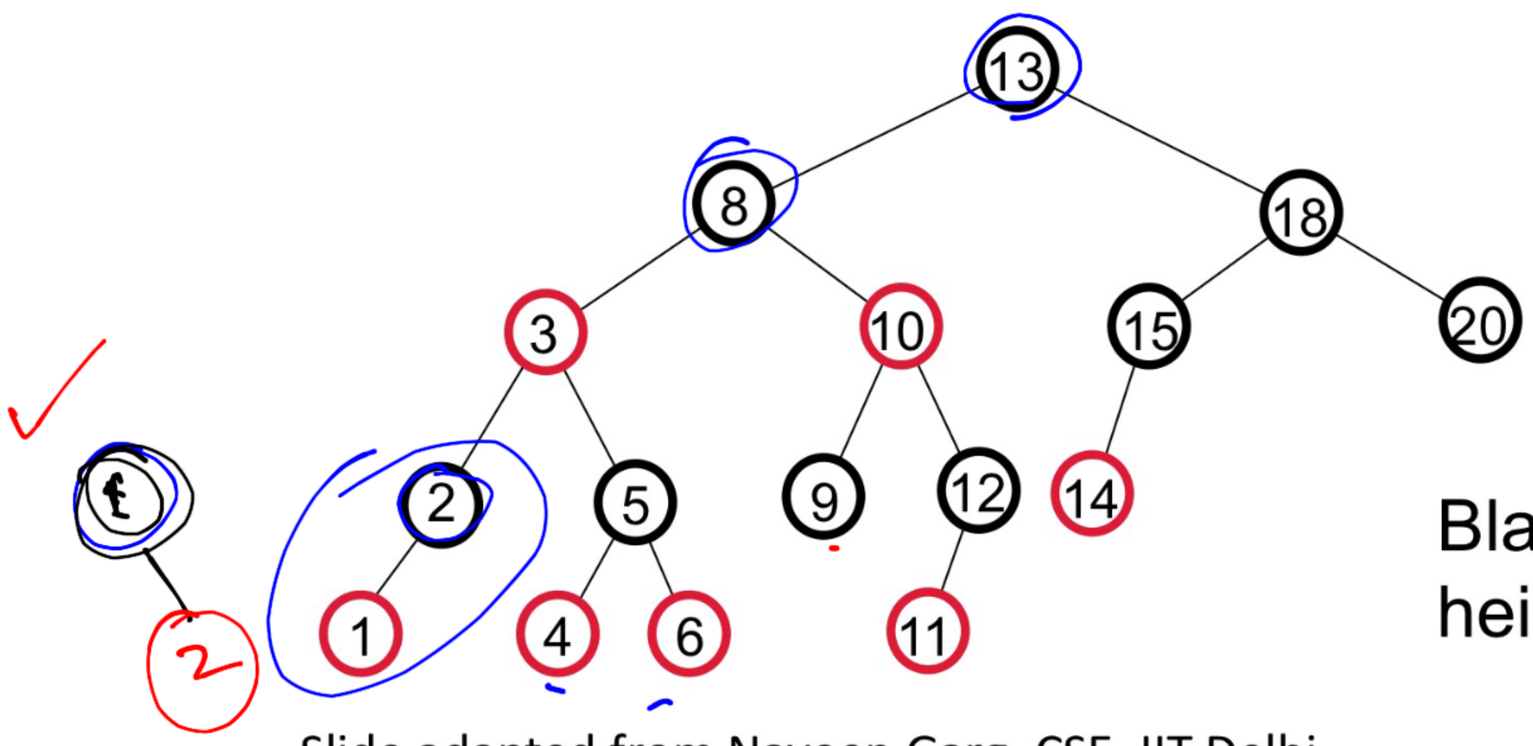
2-4 Trees to Red-Black Trees

- 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

Slide adapted from Naveen Garg, CSE, IIT Delhi

Example





Black height is same as height of 2-4 tree

Slide adapted from Naveen Garg, CSE, IIT Delhi

Find in Red-Black Trees

- Just like any other binary search trees
- O(log(n)) time

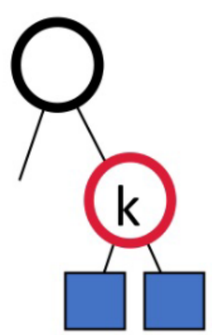
Insertion in Red-Black Trees

Insertion in Red-Black Trees

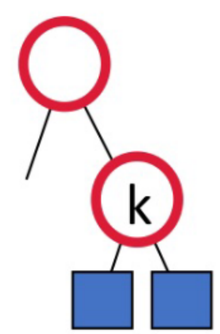
- Let k be the key being inserted
- We first search for k
- Find the place where we have to insert k
- We create a new node with key k and insert it at this place
- The new node is colored red

Insertion in Red-Black Trees

- Since inserted node is colored red, the black height of the tree remains unchanged.
- However, if the parent of inserted node is also red then we have a double red problem.



No problem

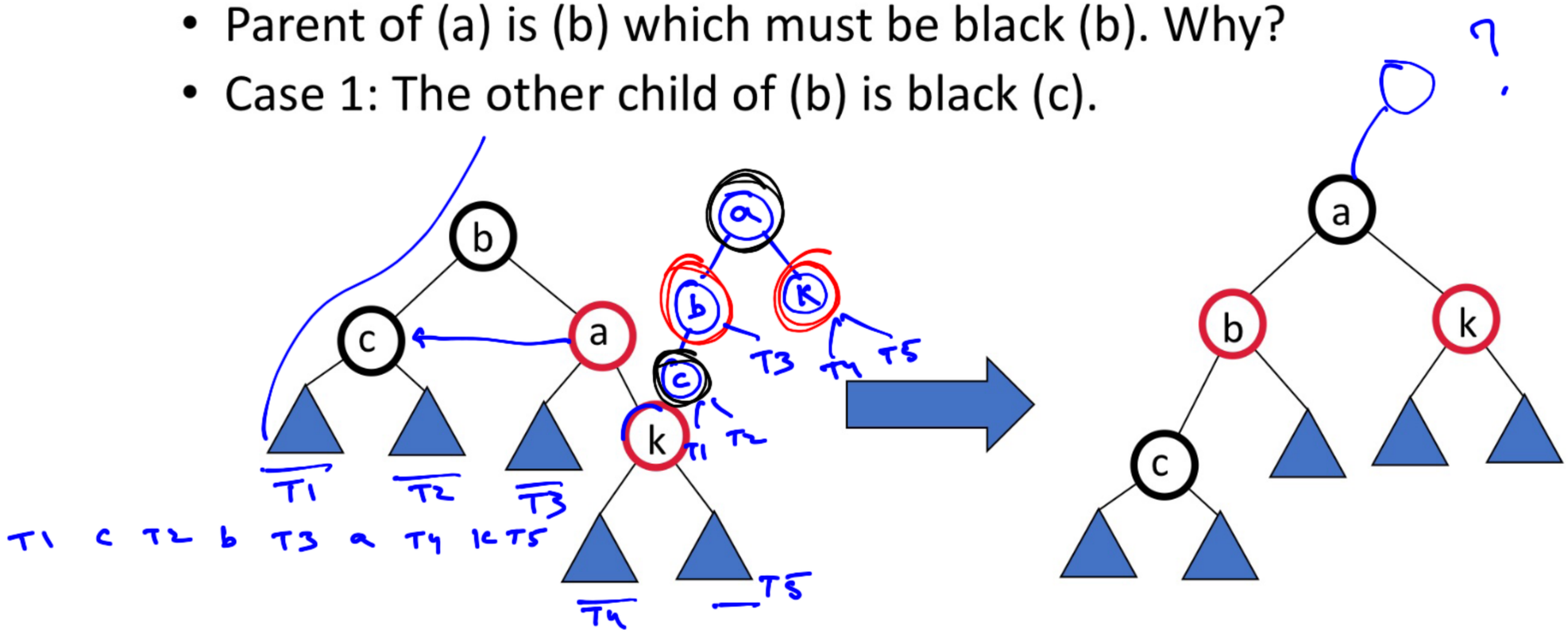


Double red problem

Slide adapted from Naveen Garg, CSE, IIT Delhi

Insertion: Case 1

- (k) is the new node inserted. (k) is colored red
- Parent of (k) is (a) which is red
- Parent of (a) is (b) which must be black (b). Why?
- Case 1: The other child of (b) is black (c).



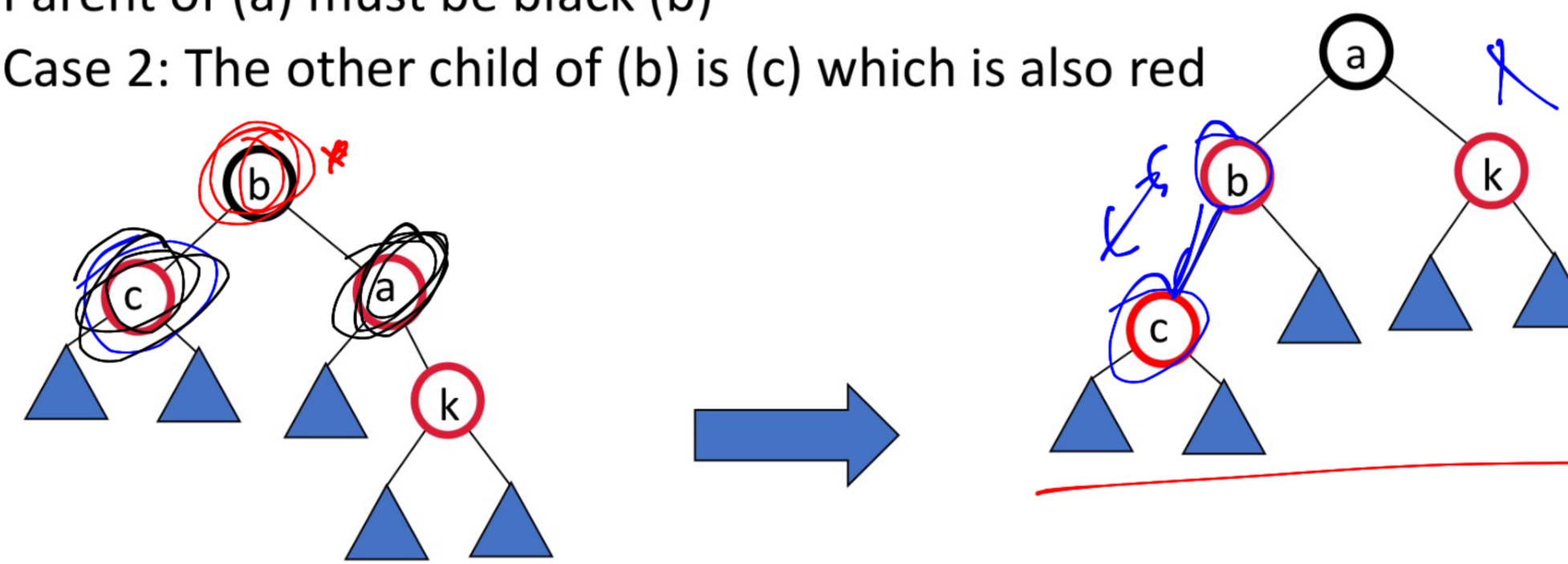
The 2-4 tree node contains {b,a,k} and is malformed.
The rotation corrects the defect.

c	**b**	**a**	**k**

Slide adapted from Naveen Garg, CSE, IIT Delhi

Insertion: Case 2

- (k) is the new node inserted. (k) is colored red
- Parent of (k) is (a) which is red
- Parent of (a) must be black (b)
- Case 2: The other child of (b) is (c) which is also red



The rotation of case 1 doesn't work in this case

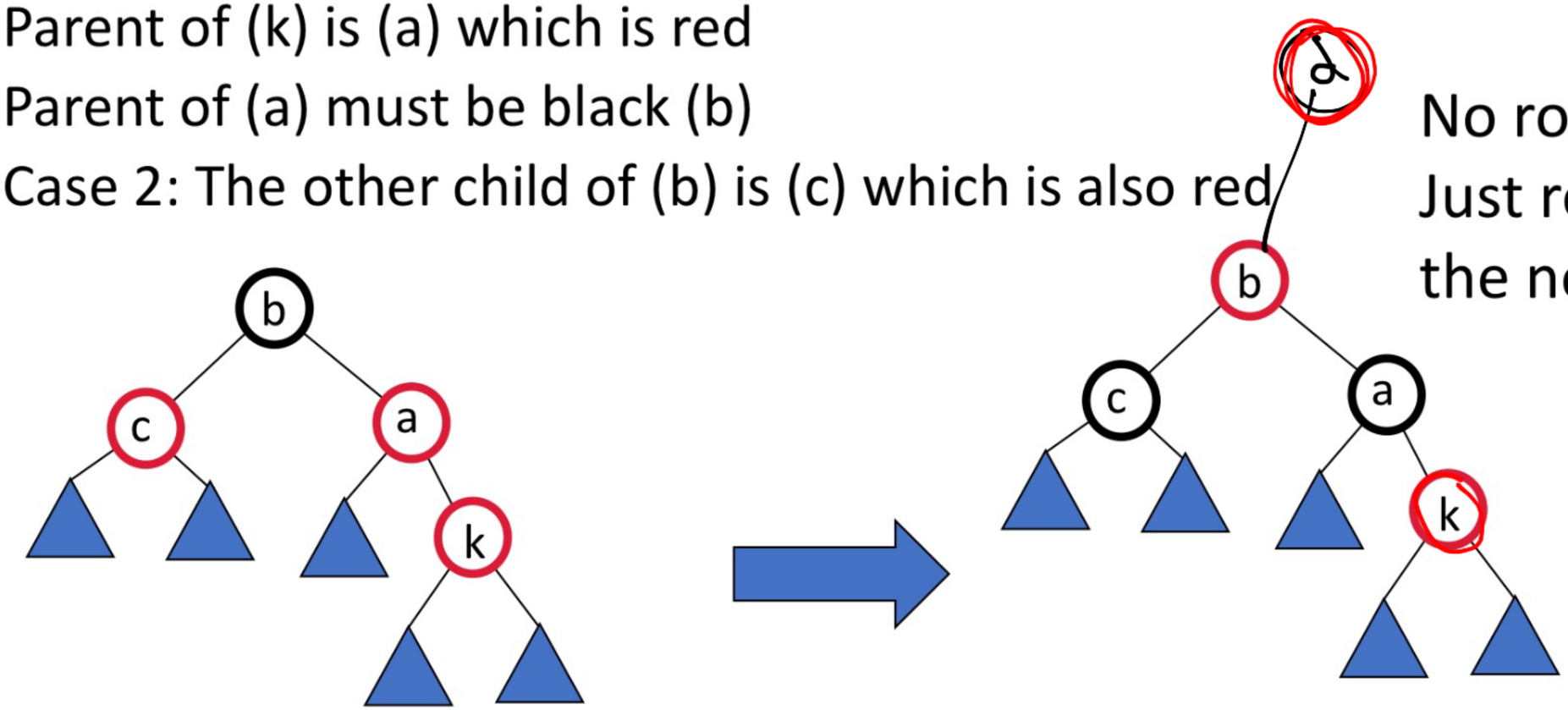
c	b	a	k

Slide adapted from Naveen Garg, CSE, IIT Delhi

Insertion: Case 2

- (k) is the new node inserted. (k) is colored red
- Parent of (k) is (a) which is red
- Parent of (a) must be black (b)
- Case 2: The other child of (b) is (c) which is also red

No rotation. Just re-color the nodes



Slide adapted from Naveen Garg, CSE, IIT Delhi

Insertion: Case 2

- The color of b changed from black to red
  - The parent of b could also be red
  - In that case, the double red problem moves up a level
- Repeat this process at the next level
- Eventually, we might need to color the root red
  - In this case we recolor the root black
  - This increases the black depth of every external node by 1
  - In the 2-4 tree this corresponds to splitting the root

Slide adapted from Naveen Garg, CSE, IIT Delhi

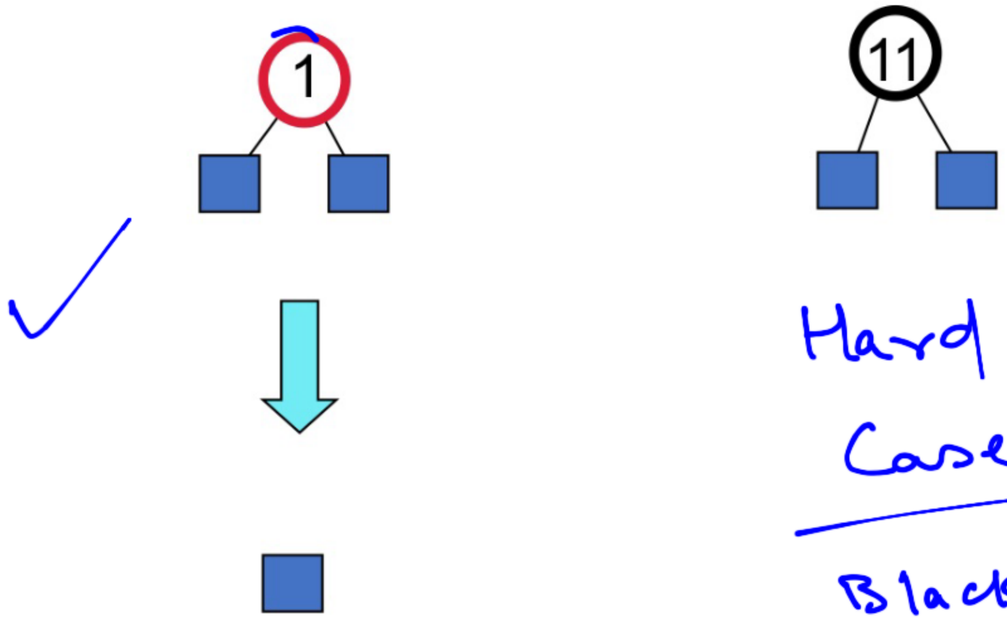
Deletion in Red-Black Trees

- Just mimic operations of a delete in a 2-4 tree
- Find the key
- Two cases
- Case 1: Key is in a leaf node
- Case 2: Key is in an internal node
  - Find predecessor of the key, swap and delete the predecessor
  - Predecessor will not have a right child
  - Therefore, one of the child of node to be deleted will be an internal node

Slide adapted from Naveen Garg, CSE, IIT Delhi

Deletion

- Thus, the node which is deleted is the parent of an external node
- Hence it is either a leaf or a non-leaf
- Case 1: Red leaf or a black leaf



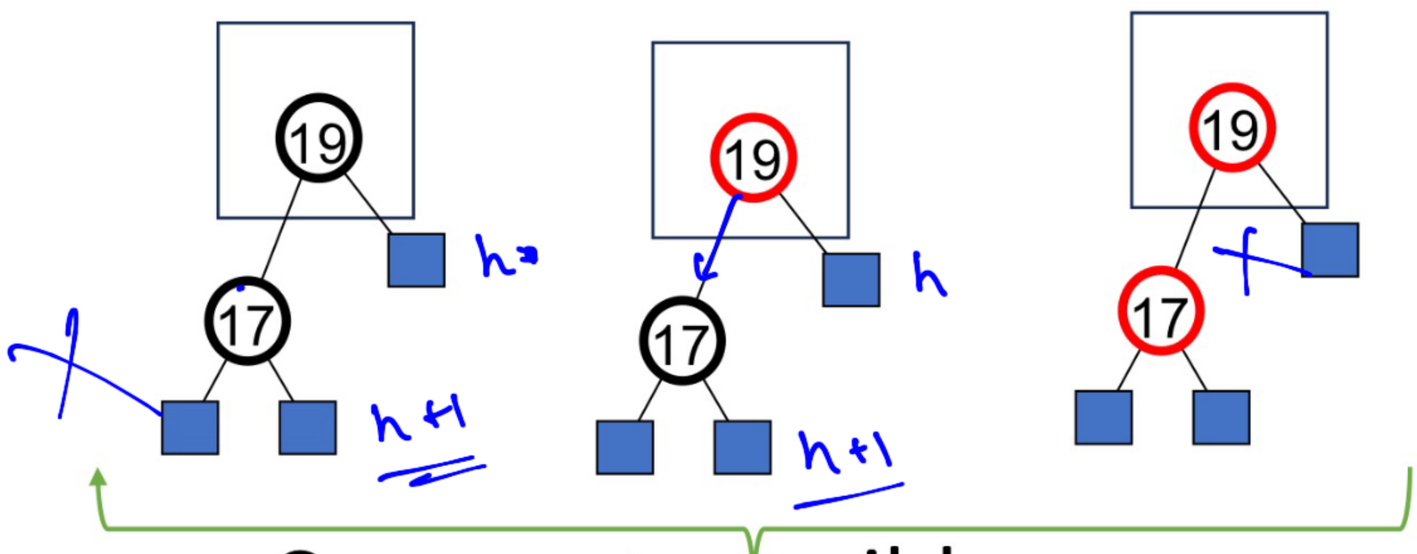
Hard Case

Black depth will change.

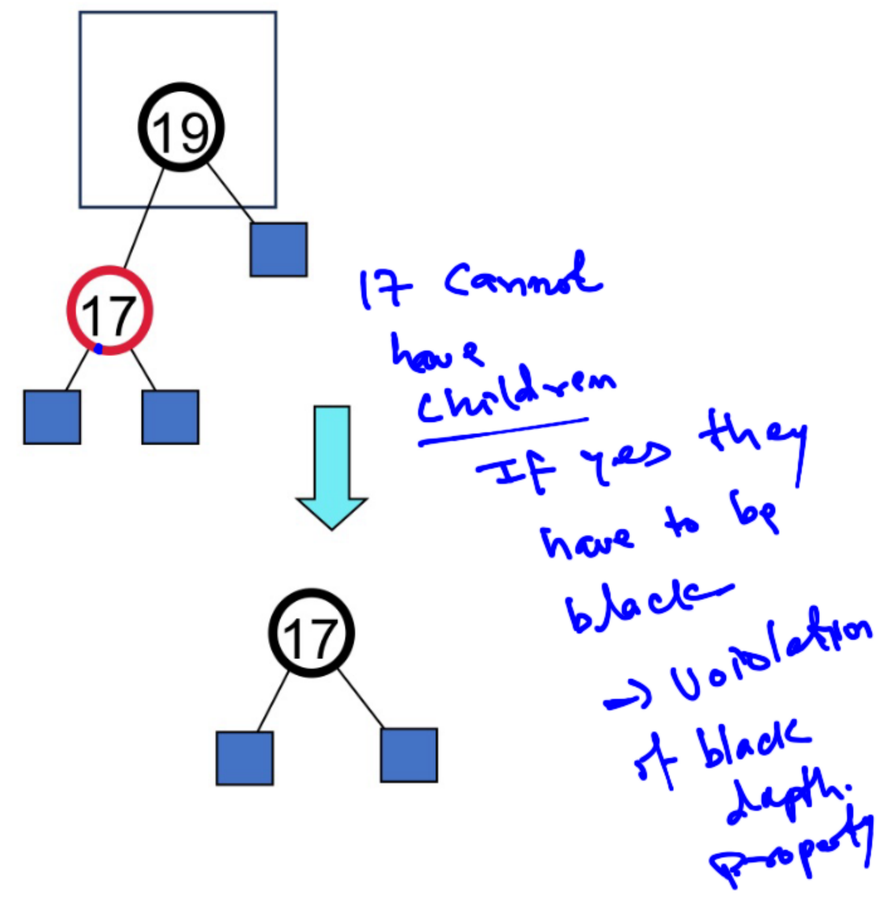
Slide adapted from Naveen Garg, CSE, IIT Delhi

Deletion

- 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
- Case 2: Non-leaf
- In this case, it needs to be a parent of a red leaf node

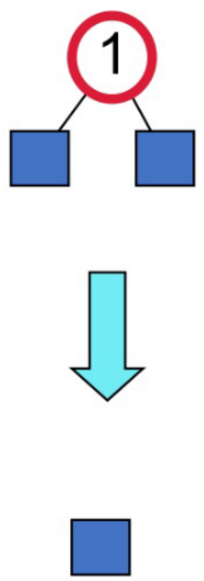


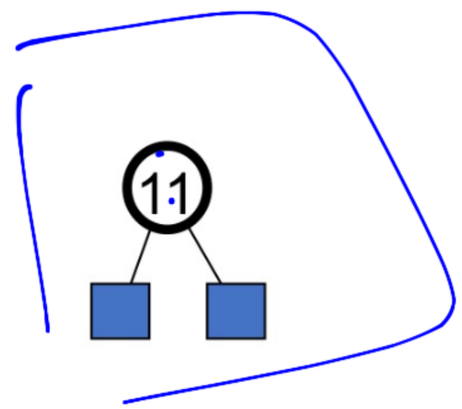
Cases not possible



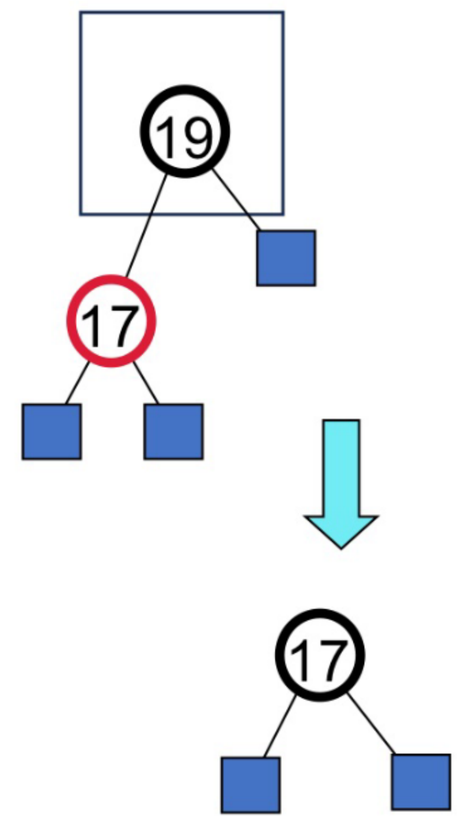
Slide adapted from Naveen Garg, CSE, IIT Delhi

Deletion





Only difficult case
Corresponds to 2-4
tree node with 1 key



Easy case

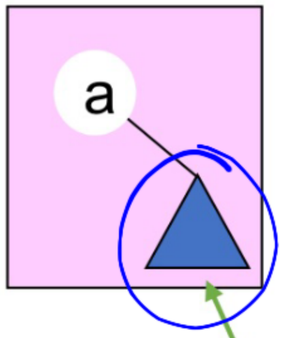
Easy case

Slide adapted from Naveen Garg, CSE, IIT Delhi

Deletion

- Assume that the node deleted is a black leaf
- Removing this reduces black depth of an external node by 1
- General step
  - Consider the largest Red-Black subtree
  - Whose black height goes down from h to h-1

Deletion: Cases

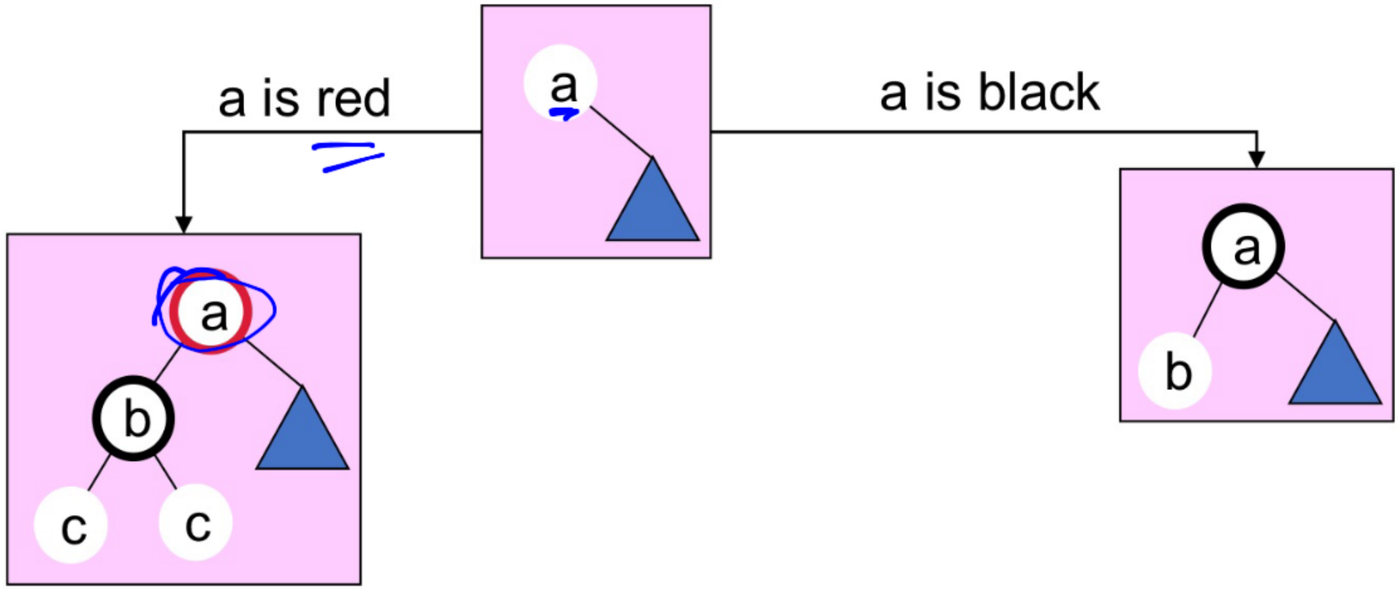


Largest red-black subtree T whose

black height has reduced from h to h-1

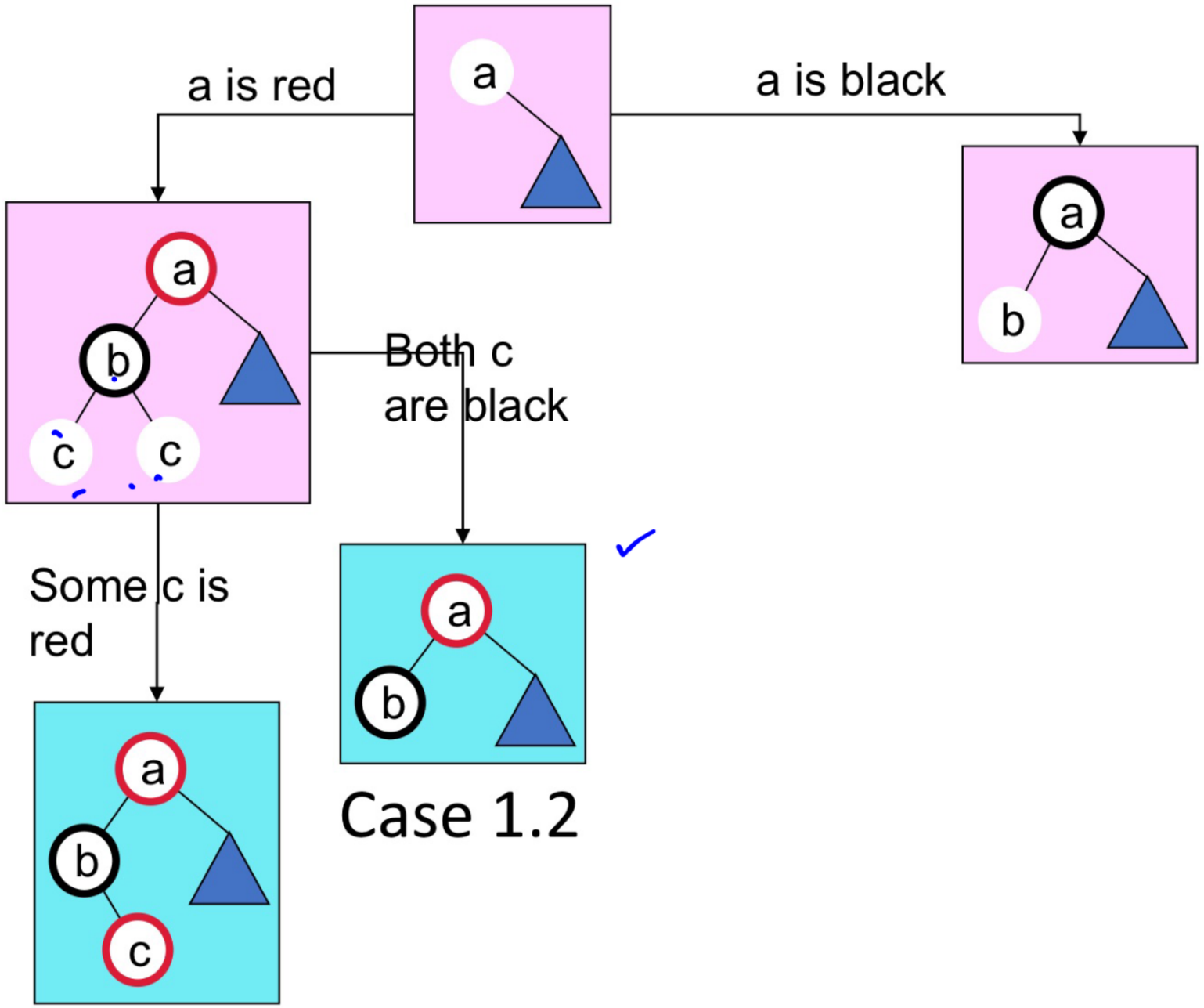
Slide adapted from Naveen Garg, CSE, IIT Delhi

Deletion: Cases



Slide adapted from Naveen Garg, CSE, IIT Delhi

Deletion: Cases

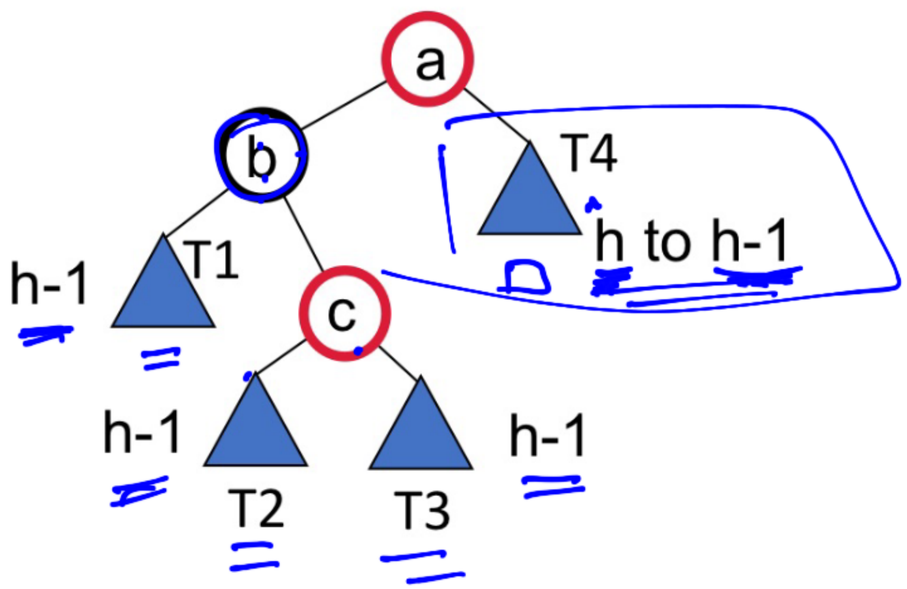


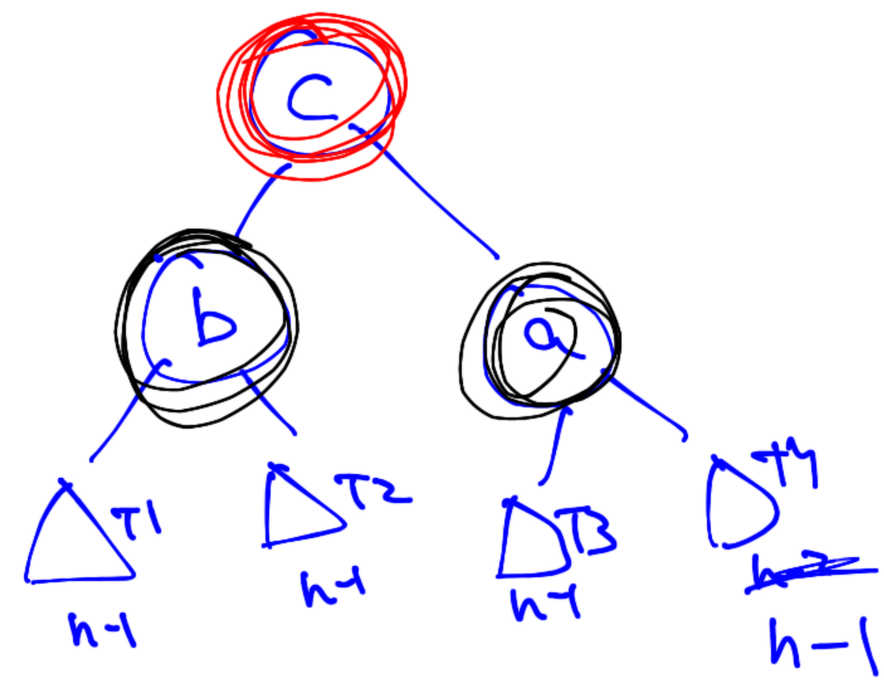
Case 1.1

Slide adapted from Naveen Garg, CSE, IIT Delhi

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)



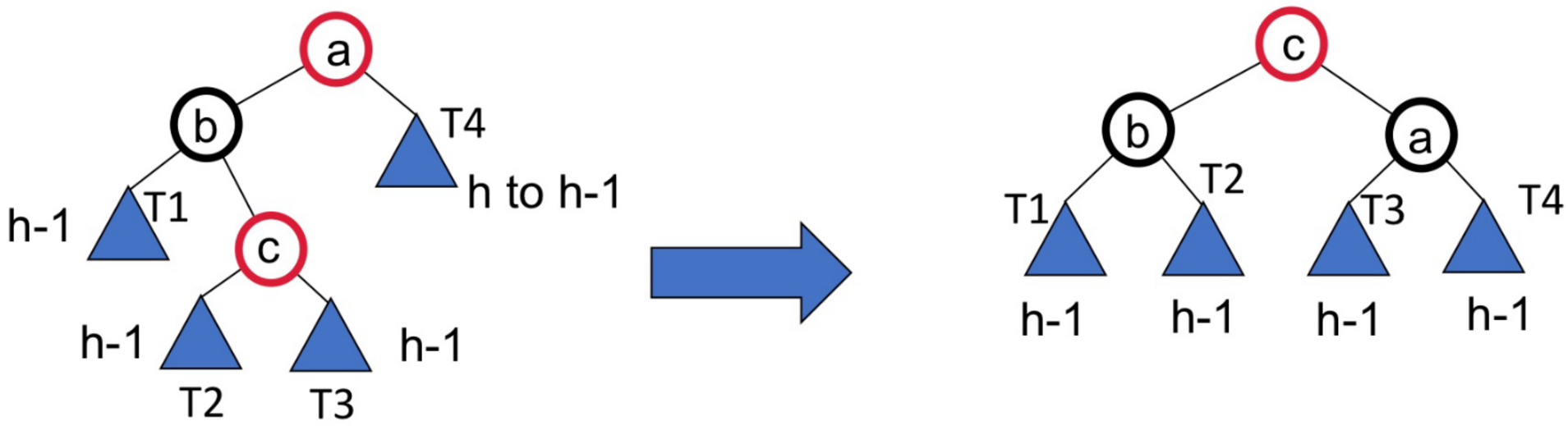


T1, b, T2, c, T3, a, T4

Slide adapted from Naveen Garg, CSE, IIT Delhi

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)



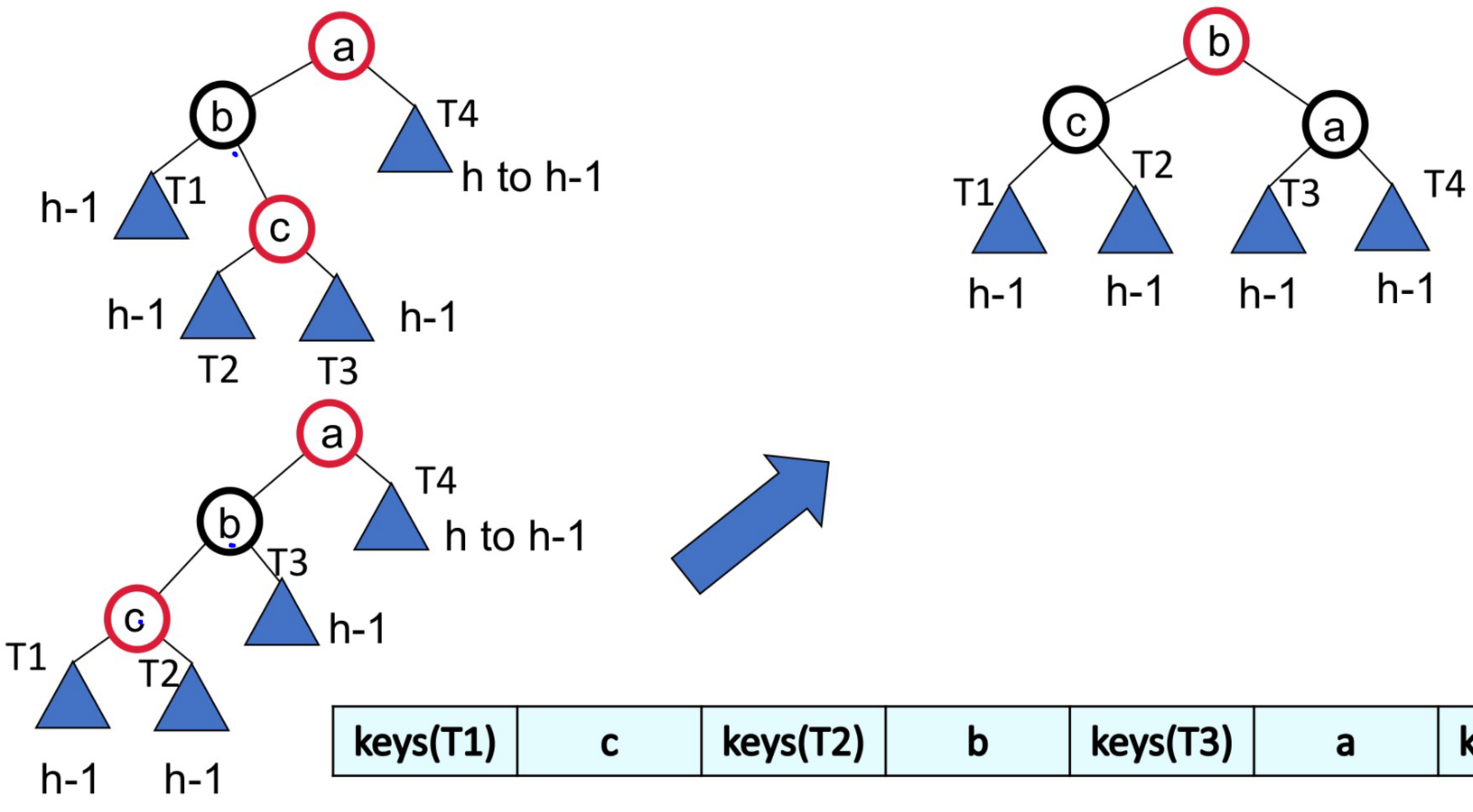
keys(T1)	b	keys(T2)	c	keys(T3)	a	keys(T4)

Slide adapted from Naveen Garg, CSE, IIT Delhi

Slide adapted from Naveen Garg, CSE, IIT Delhi

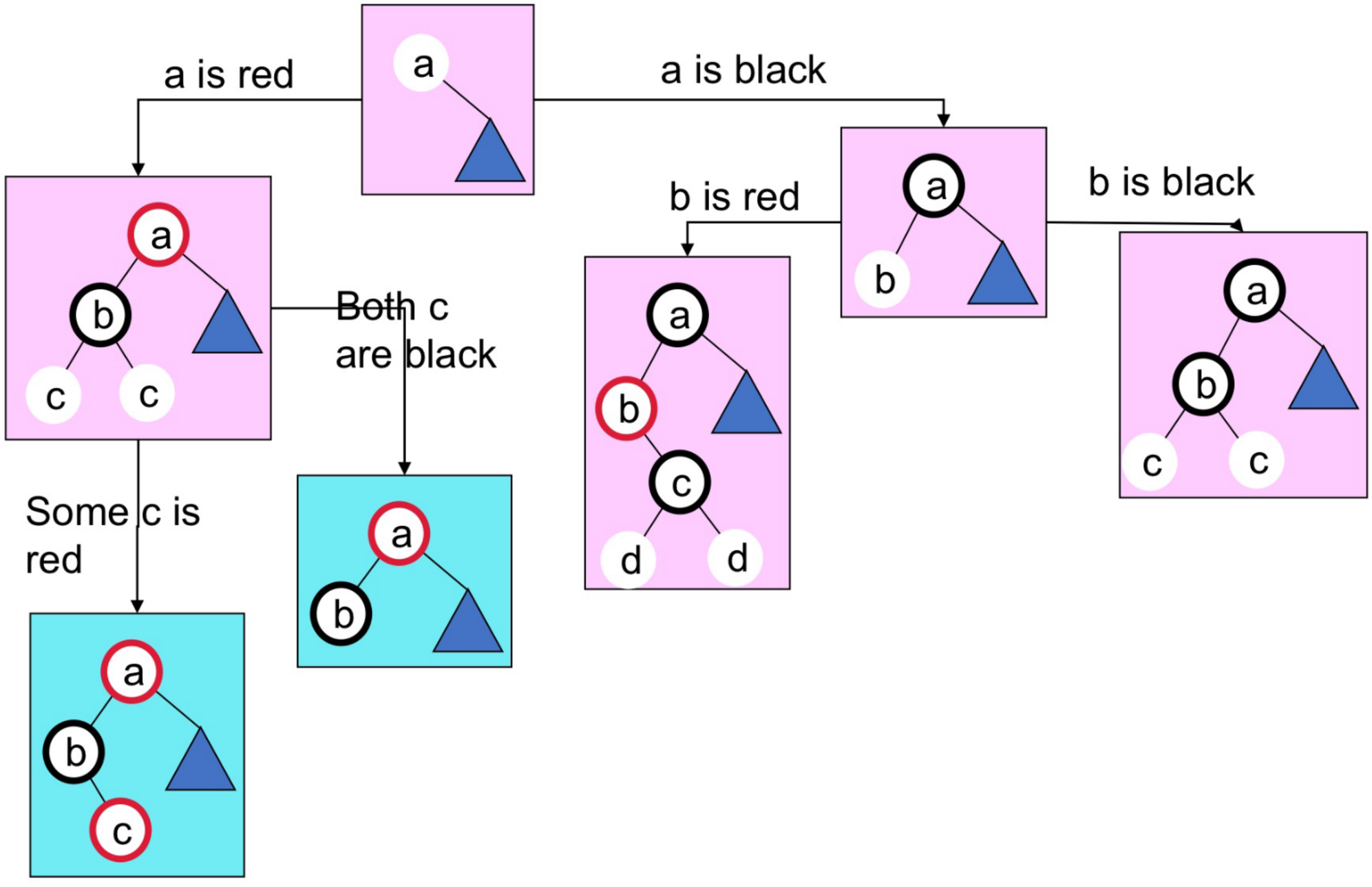
Deletion: Case1.1

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



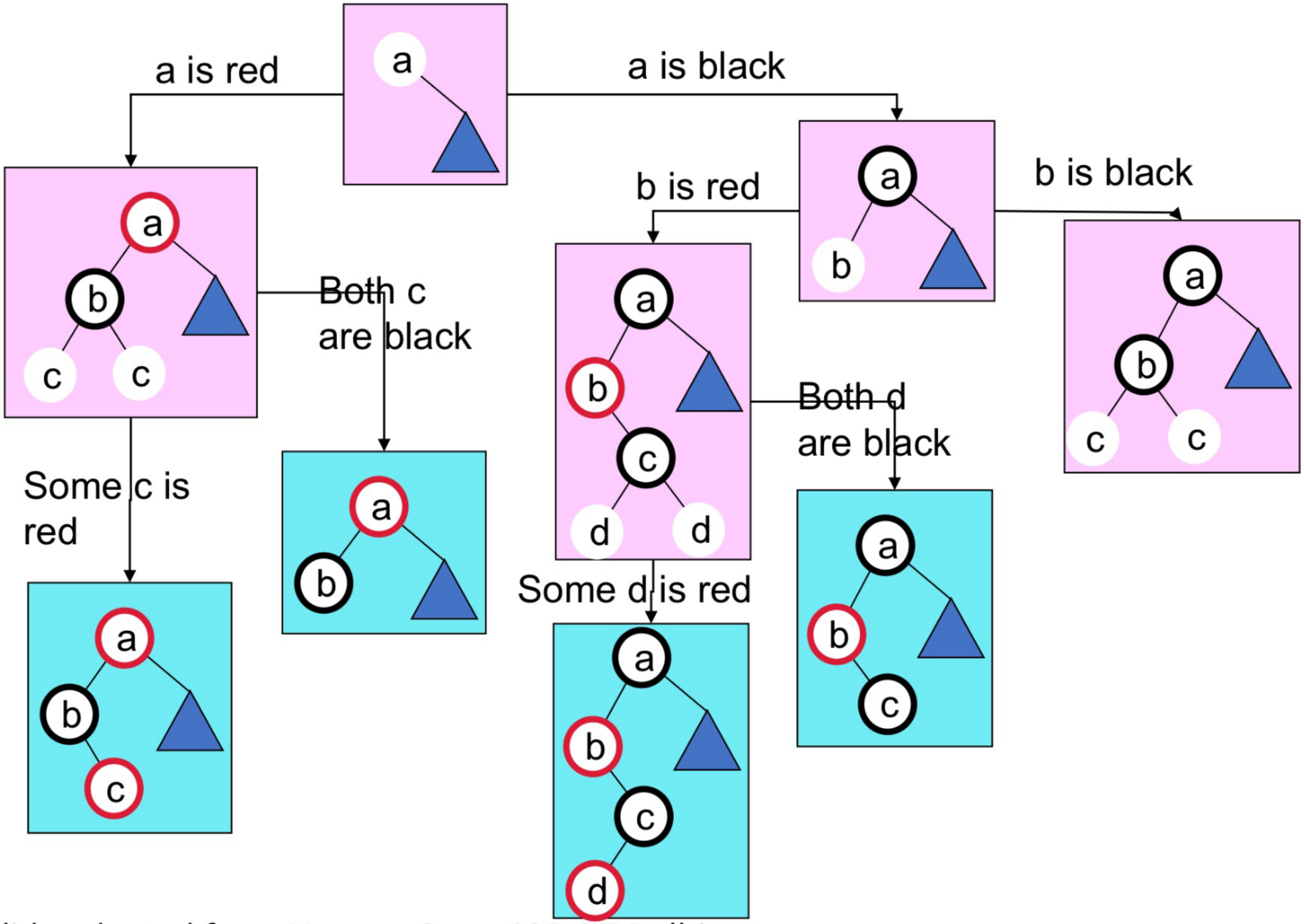
keys(T1)	c	keys(T2)	b	keys(T3)	a	keys(T4)

Deletion: Cases



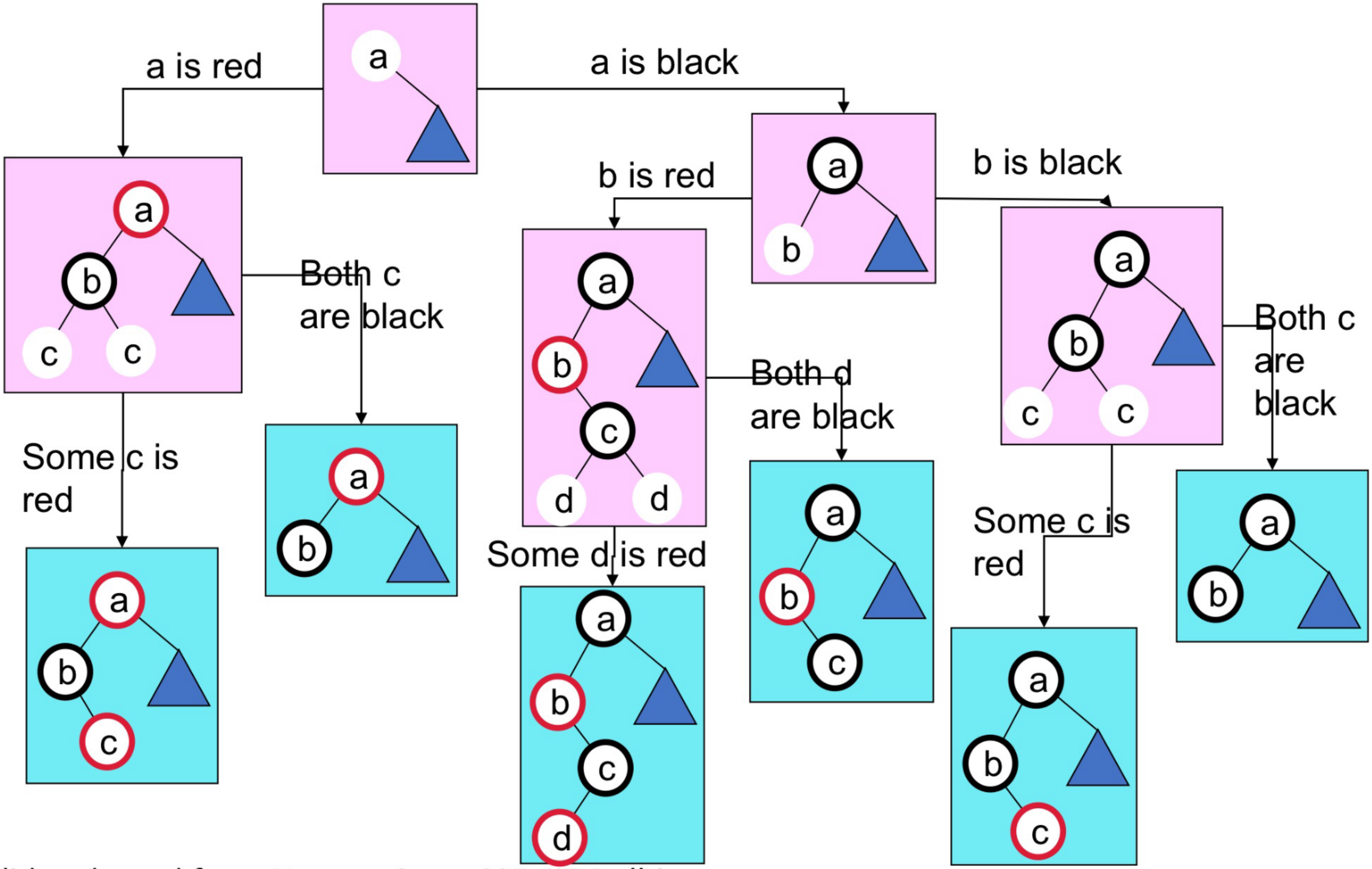
Slide adapted from Naveen Garg, CSE, IIT Delhi

Deletion: Cases



Slide adapted from Naveen Garg, CSE, IIT Delhi

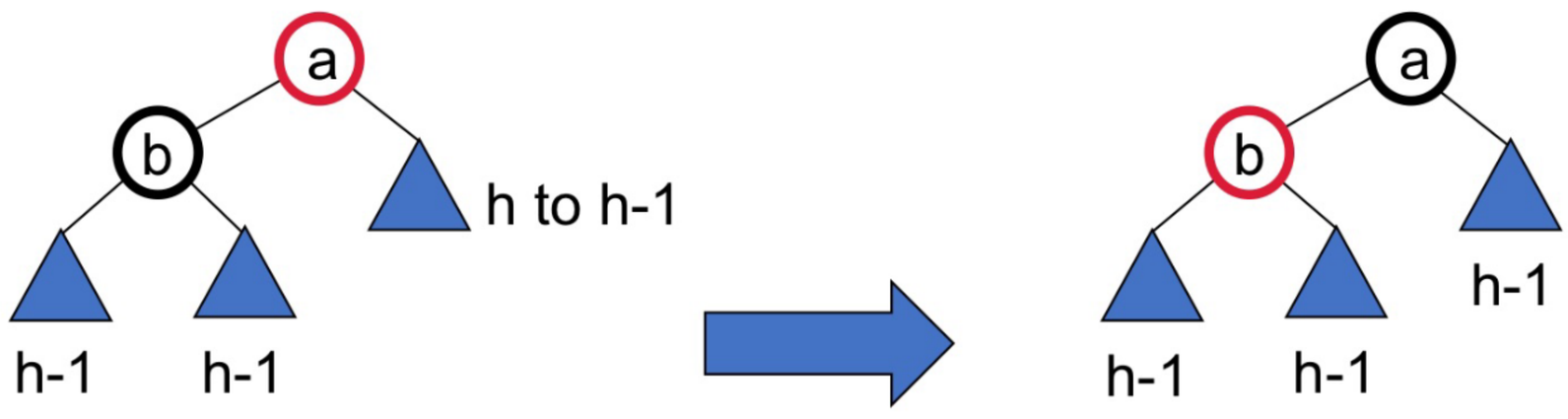
Deletion: Cases



Slide adapted from Naveen Garg, CSE, IIT Delhi

Deletion: case 1.2

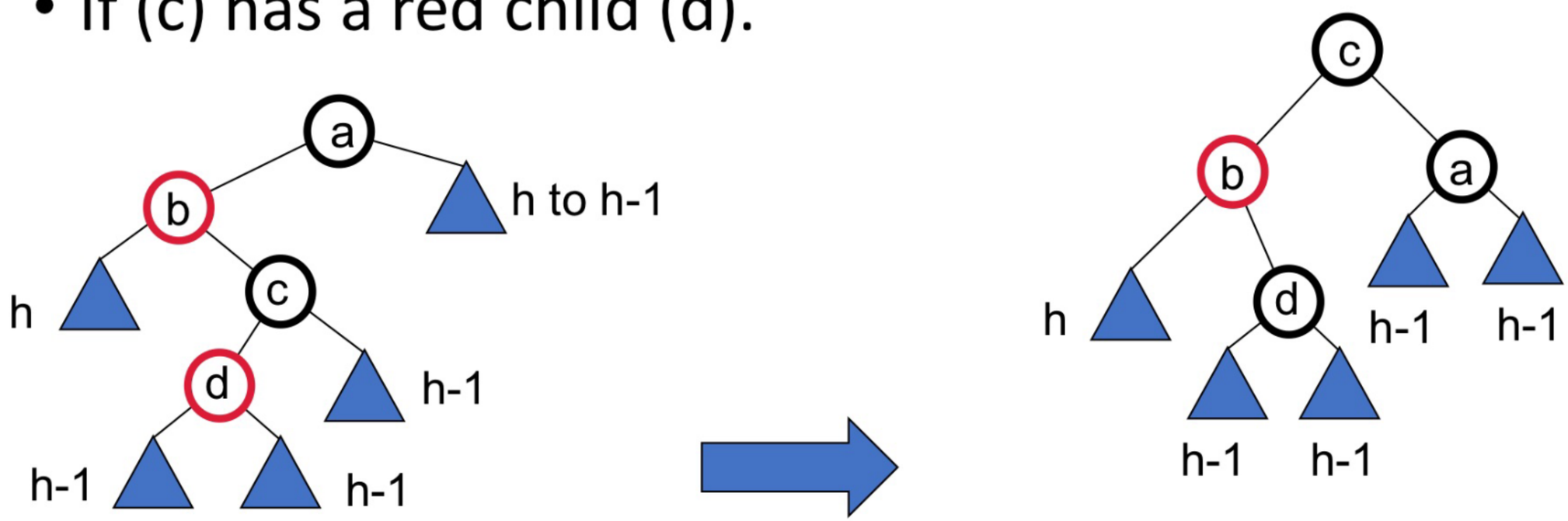
- If parent is a red node (a).
- Then it has a child (b) which must be black
- If b has no red child



Slide adapted from Naveen Garg, CSE, IIT Delhi

Deletion: case 2.1.1

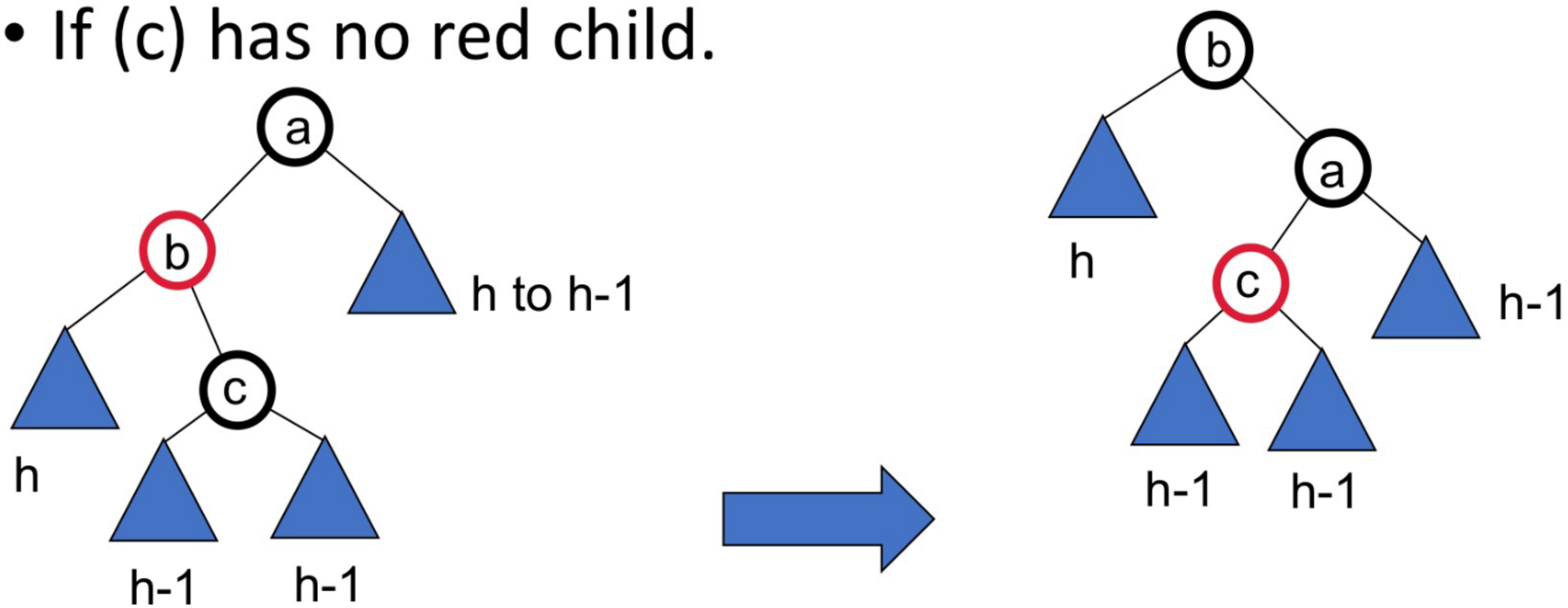
- If parent is a black node (a).
- If a has a red child (b)
- Consider right child of b (c) which must be black.
- If (c) has a red child (d).



Slide adapted from Naveen Garg, CSE, IIT Delhi

Deletion: case 2.1.2

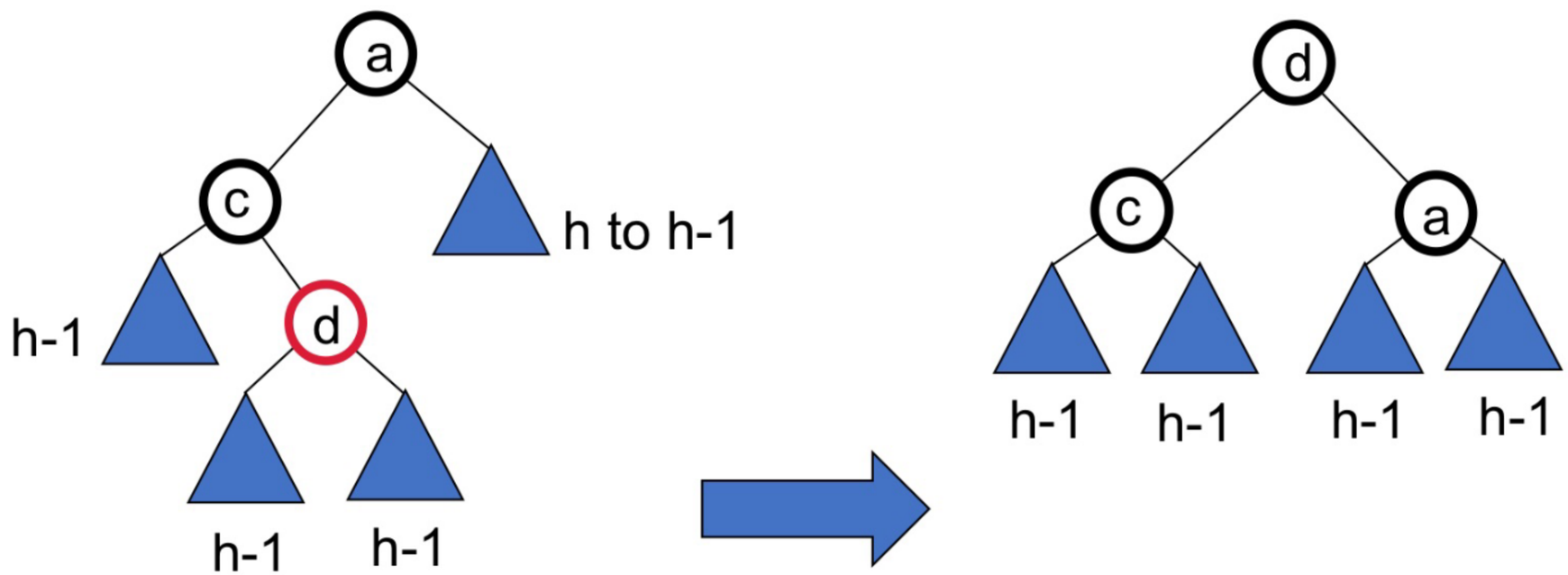
- If parent is a black node (a).
- If a has a red child (b)
- Consider right child of b (c) which must be black.
- If (c) has no red child.



Slide adapted from Naveen Garg, CSE, IIT Delhi

Deletion: case 2.2.1

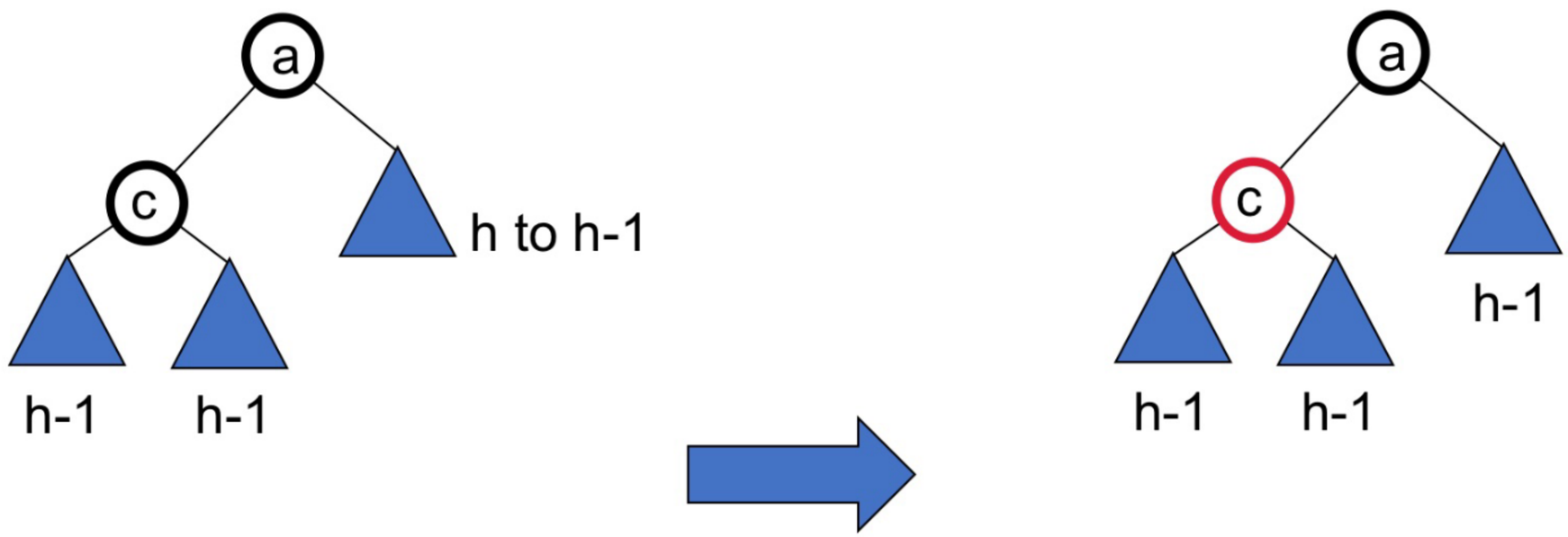
- If parent is a black node (a).
- If the child of node a (c) is black.
- If (c) has a red child (d)



Slide adapted from Naveen Garg, CSE, IIT Delhi

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.



Slide adapted from Naveen Garg, CSE, IIT Delhi

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.

Slide adapted from Naveen Garg, CSE, IIT Delhi

Insertion and Deletion: Summary

- In both insertion and deletion we need to make at most one rotation.
- We might have to move up the tree but in doing so we only recolor nodes.
- Time taken is O(log n)

Slide adapted from Naveen Garg, CSE, IIT Delhi

Thank You

$$((x + y) + z) = (x + (y + z)) \ ?$$

find x: $f(x) = 0$

$$\text{abs}(f(x)) < \epsilon$$

$$\boxed{x = q \ ?}$$