Data Structures and Algorithms (11)

Instructor: Ming Zhang
Textbook Authors: Ming Zhang, Tengjiao Wang and Haiyan Zhao
Higher Education Press, 2008.6 (the "Eleventh Five-Year" national planning textbook)

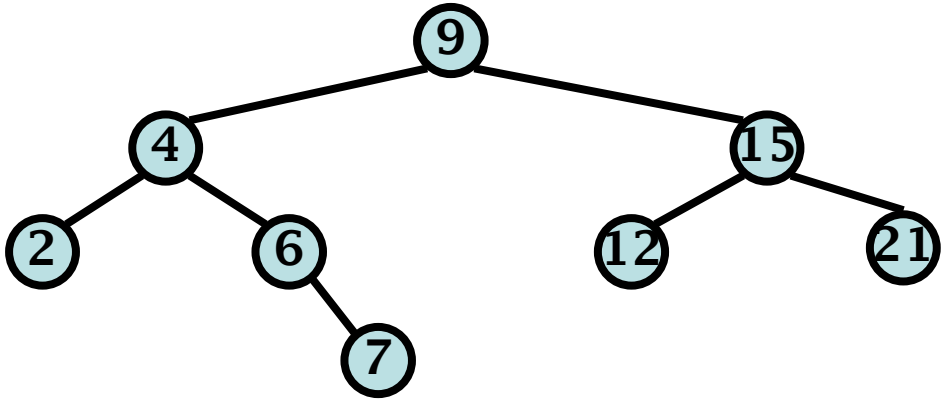
https://courses.edx.org/courses/PekingX/04830050x/2T2014/

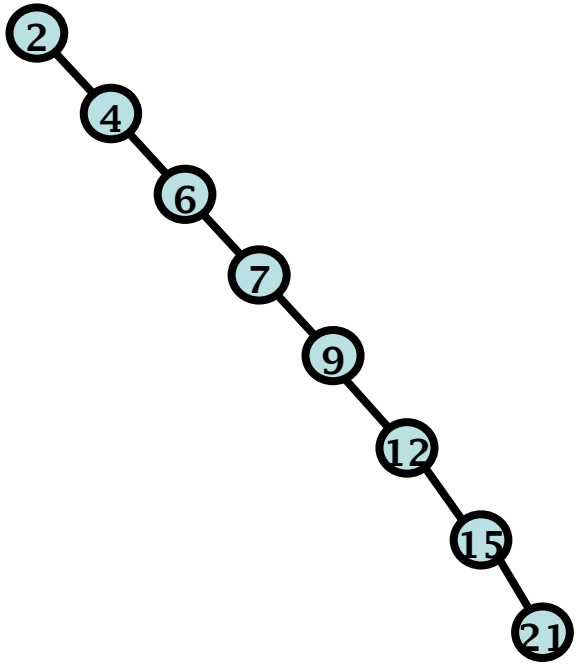
Content

- Basic Concepts
- 11.1 Linear Indexing
- 11.2 Static Indexing
- 11.3 Inverted Indexing
- 11.4 Dynamic Indexing
- 11.5 Bit Indexing
- 11.6 Red-Black Tree

Balance Problem of BST

- Ideal case: cost of insertion, deletion, search of an element is O(log$n$)
- input 9, 4, 2, 6, 7, 15, 12, 21
- output 2, 4, 6, 7, 9, 12, 15, 21





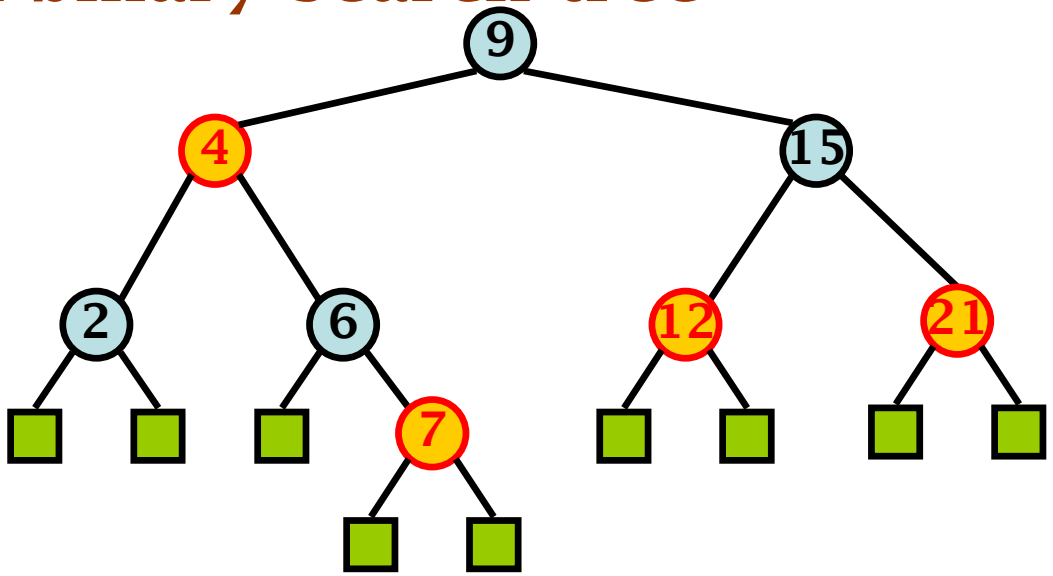
11.6 Red-Black Tree

- 11.6.1 Definition

  red-black tree, or RB-tree for short
- 11.6.2 Properties of RB-tree
- 11.6.3 Algorithm for insertion
- 11.6.4 Algorithm for deletion

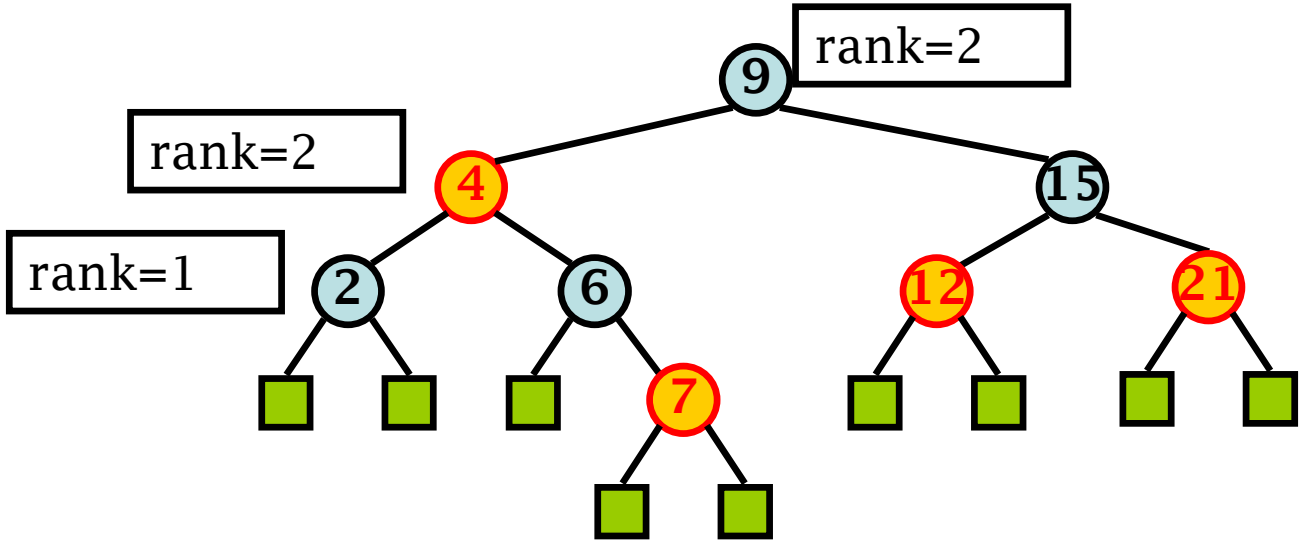
RB-Tree: balanced, extended binary search tree

- Color property: each node is either red or black;
- Root property: the root is always black;
- External nodes property: extended external leaves are all null black nodes;
- Internal nodes property: children of a red node are all black; no consecutive red nodes are allowed;
- Path property: each path starting from the root to an external leaf contains the same number of black nodes.



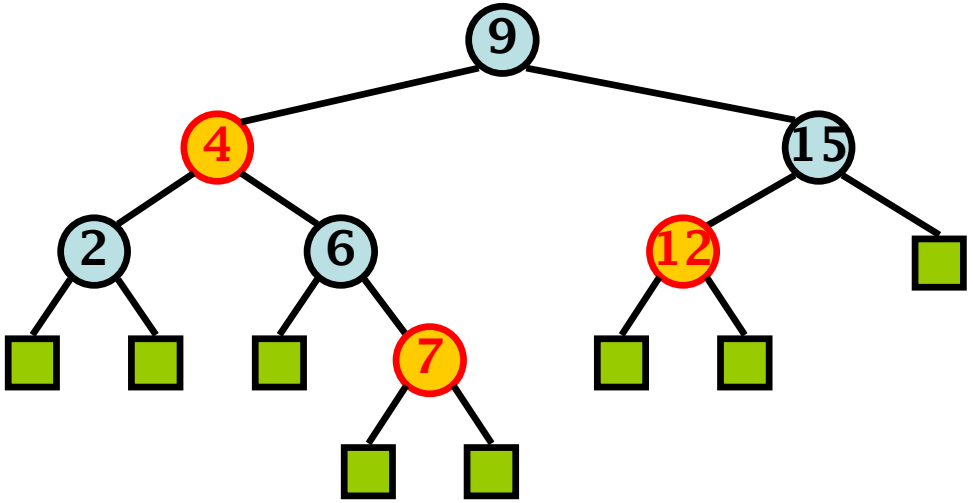
Rank of RB-Tree

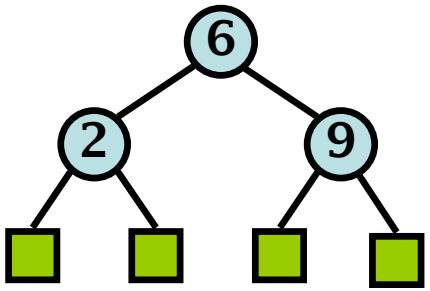
- Rank of a node X, (also called the black height)
  - The number of black nodes from it to an external node.
  - X itself is not included, while the leaf node is included.
- Rank of an external node is zero
- Rank of the root is the rank of the tree



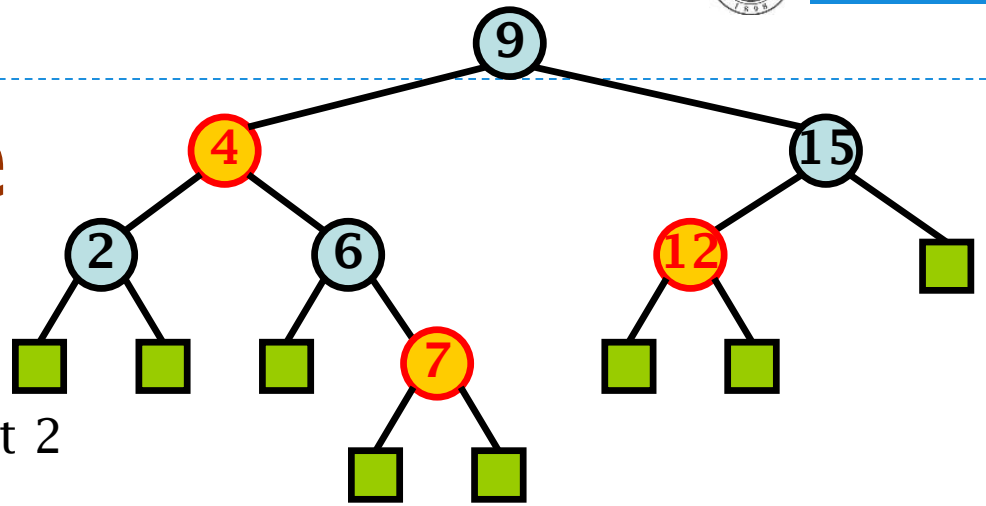
11.6.2 Properties of RB-Tree

- (1) RB-tree is a full binary tree, when null leaves are viewed as nodes.
- (2) The length of a simple path from the root to a leaf of a k-rank RB-tree ranges from k to 2k





Properties of RB-Tree



- (2)' Height of a k-rank RB-tree ranges from k+1 to 2k+1
- (3) There are at least 2^k-1 internal nodes in an RB-Tree
- (4) The height of a RB-tree with n internal nodes is at most 2 $\log_2$ (n+1) +1

- Proof:

  Suppose the rank is k and the height is h

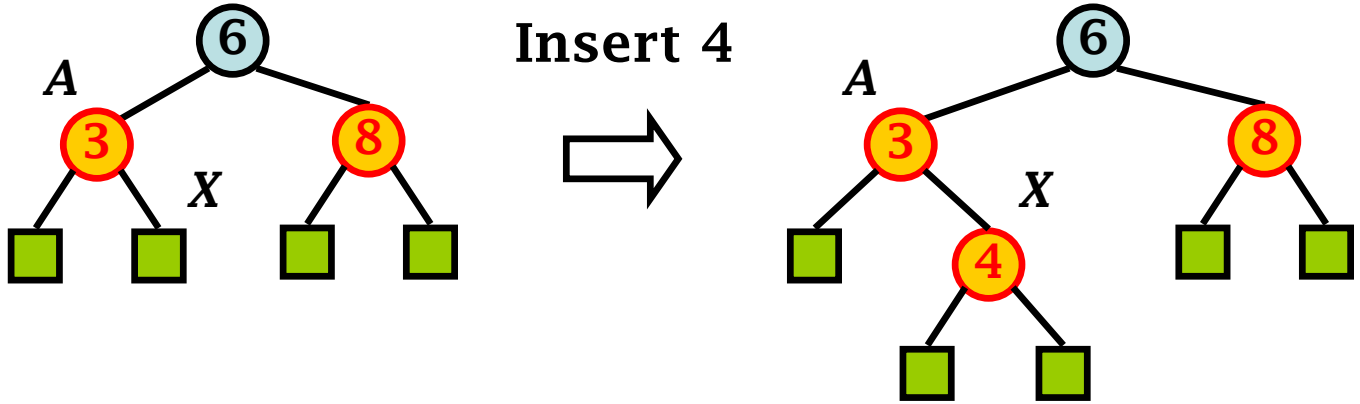
  from property(2)', h <= 2k+1, so k >= (h-1) / 2

  from property(3), n >= 2^k – 1, so>= $2^{(h-1)/2}$ – 1

  and thus h <= 2 $\log_2$ (n+1)+1

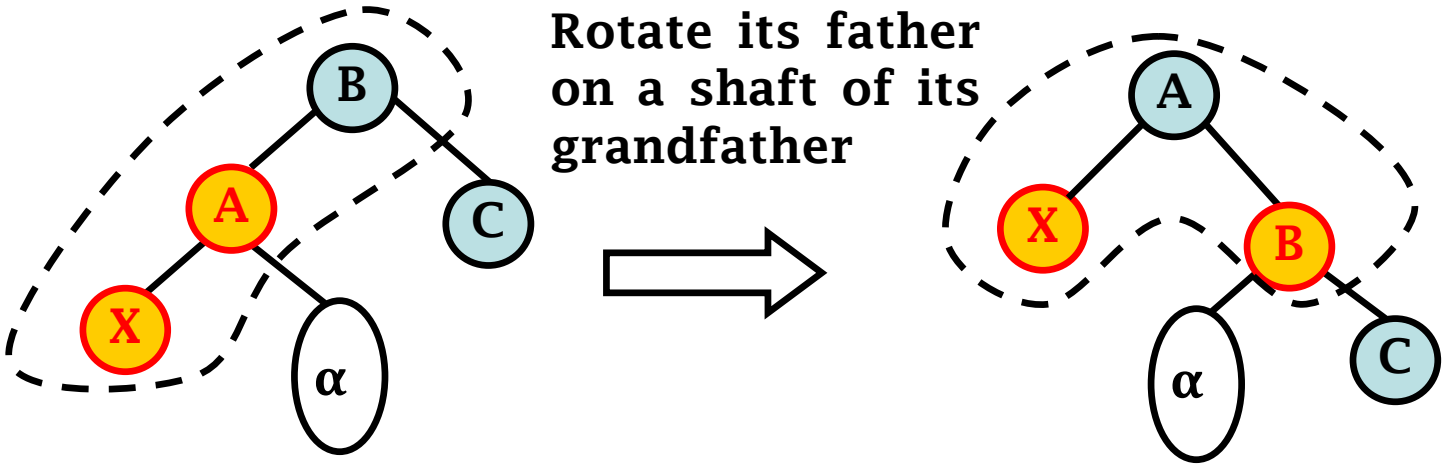
11.6.3 Algorithm for Insertion

- First call the insertion algorithm for BST
  - Color the new record red
  - If its father is black then the insertion is completed
- Otherwise, perform the double red adjustment.



Adjustment for Insertiong: Case 1 – Rotation

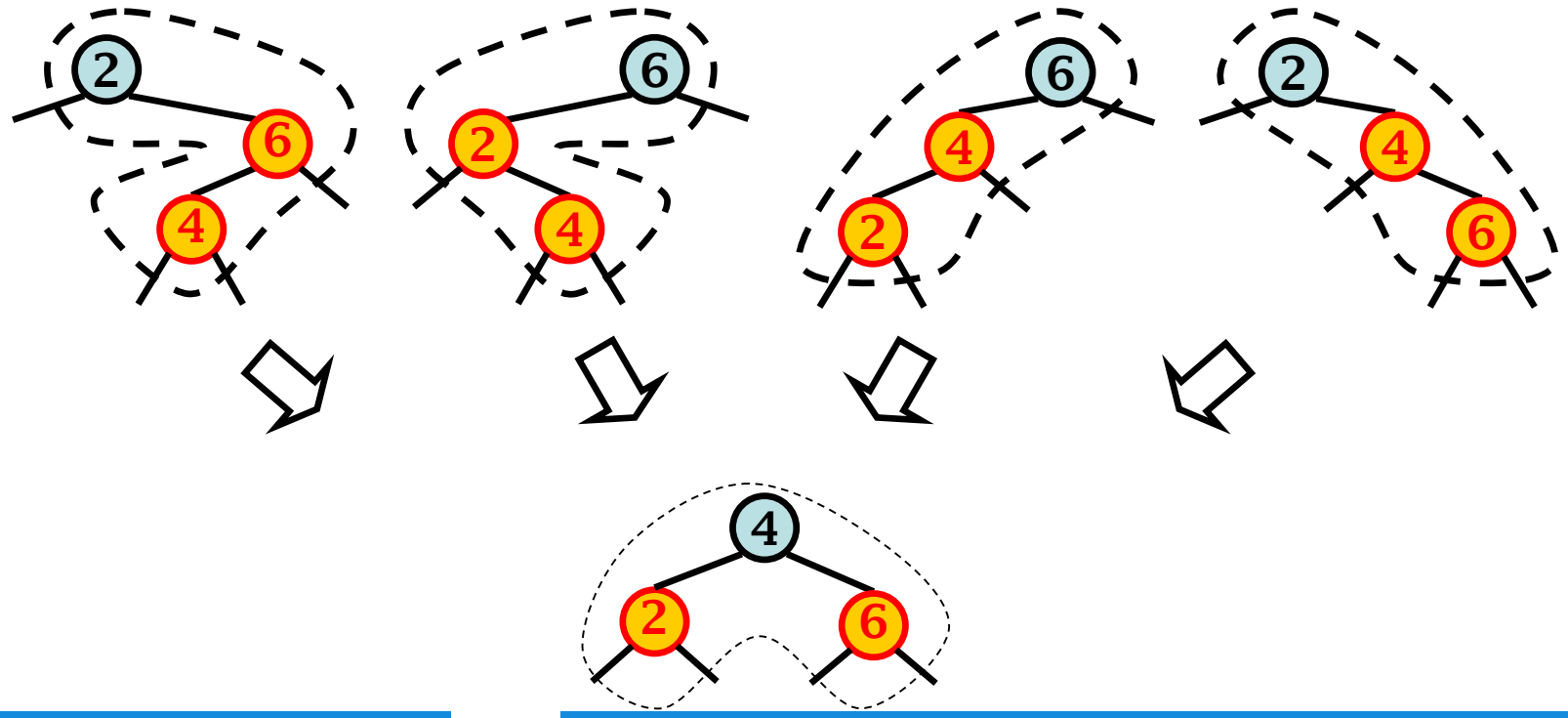
- Case1: The uncle of X is black



- Rank of each node remains the same, adjustment is completed

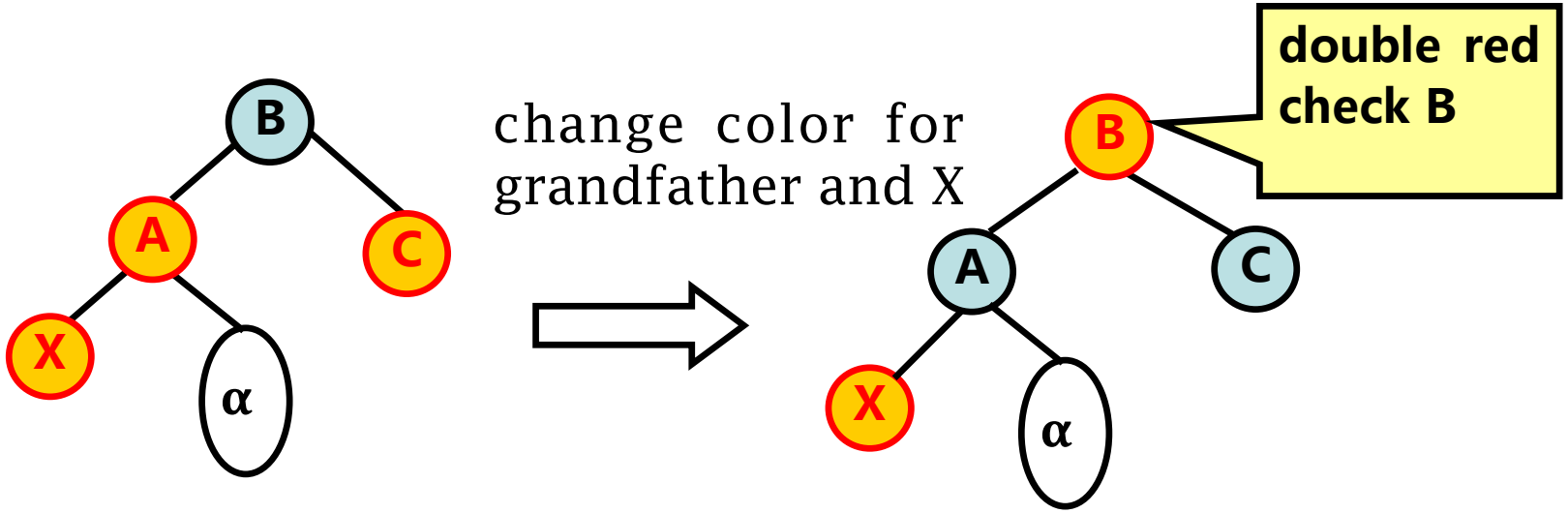
4 types of structure adjustment

- Principle: Maintaining the in-order property of BST



Adjustment for Insertion: case 2 – re-coloring

- Case 2: the uncle of X is also red

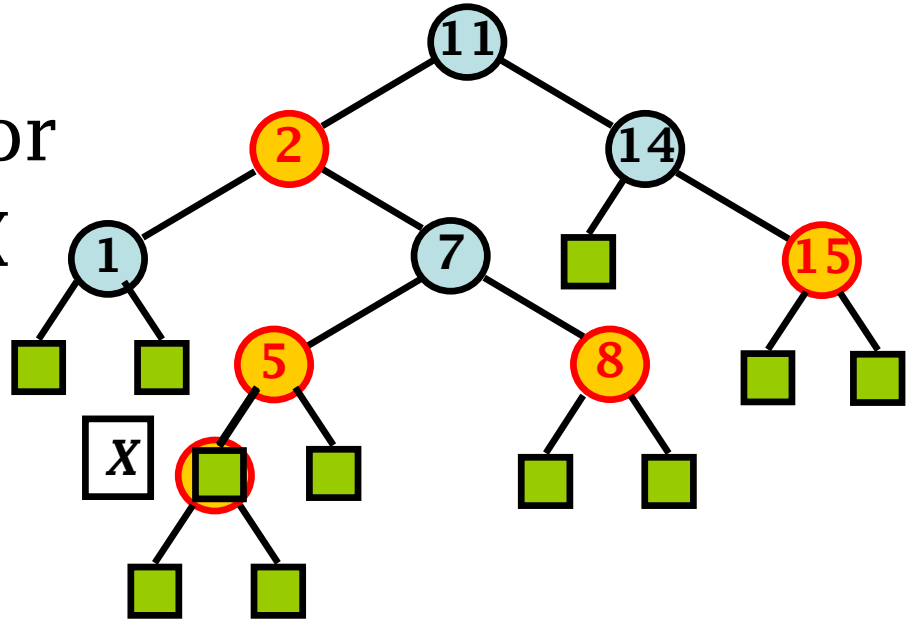


- requires further balance check

Insert 4

- Case 2 double red conflict
  - father and uncle are both red

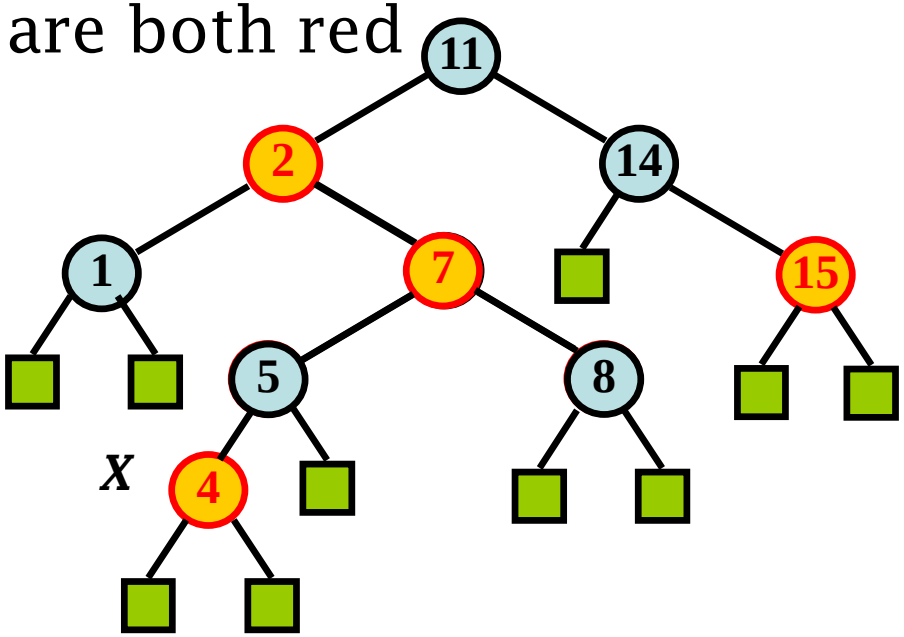
- change colors for grandfather and X



Insert 4

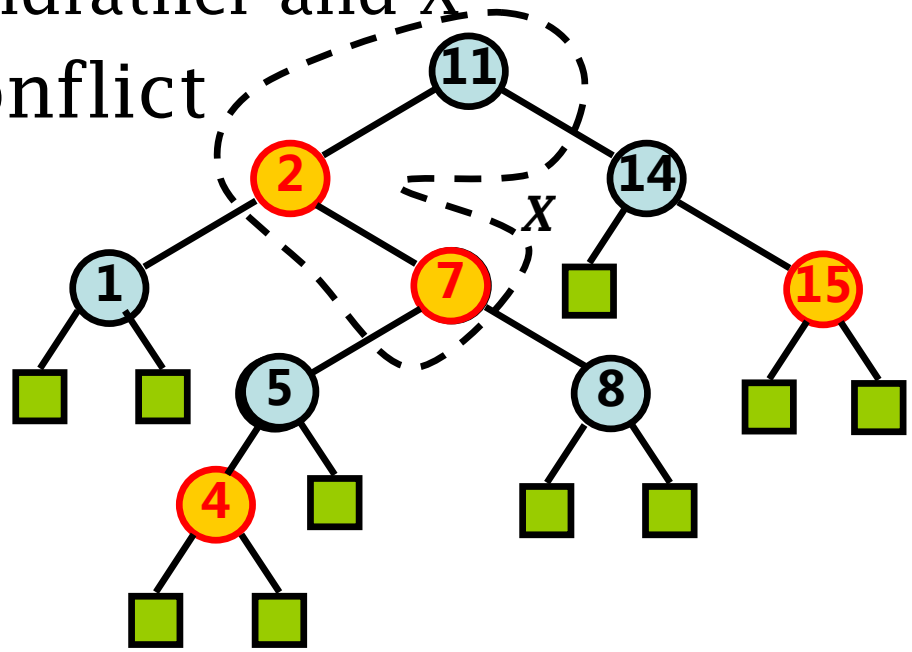
- Case 2 double red conflict
  - father and uncle are both red

- change colors for grandfather and X



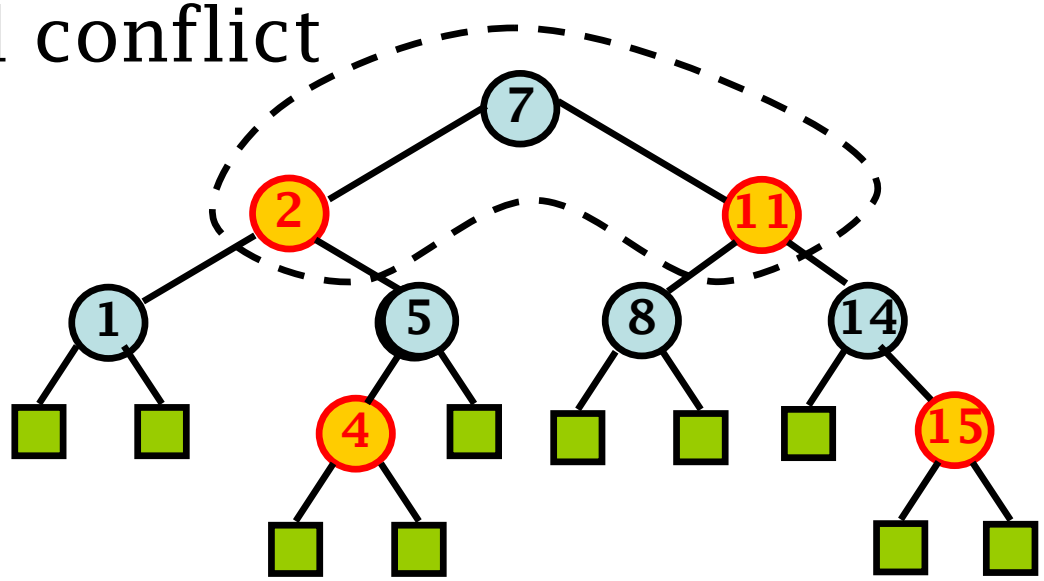
Insert 4

- Case 2 double red conflict
- Father and uncle are both red
  change colors for grandfather and X
- Case 1 double red conflict
  - Uncle is black
- Reconstruction



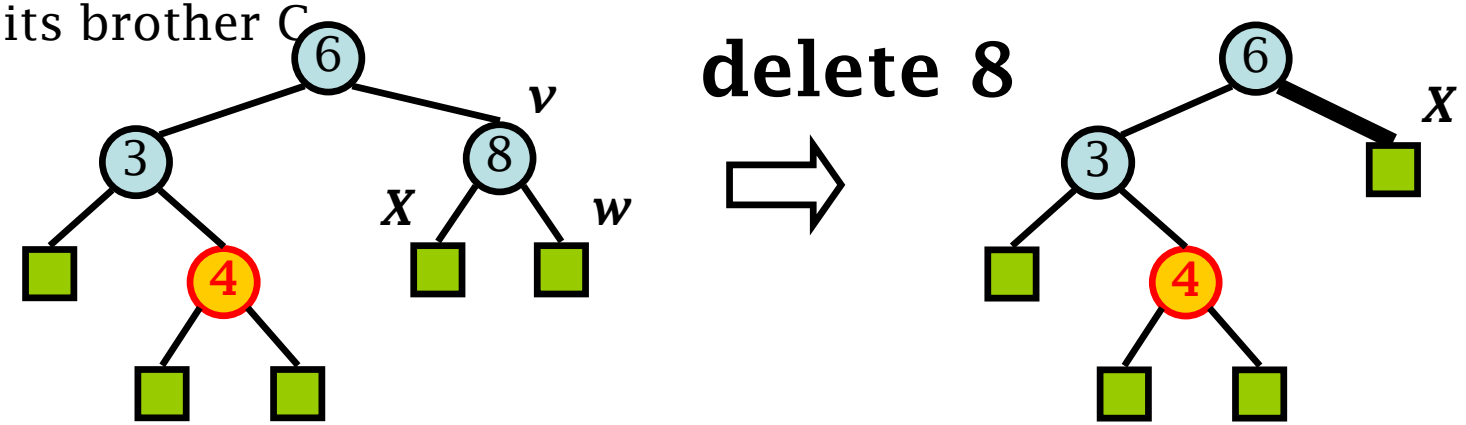
Insert 4

- Case 2 double red conflict
- Father and uncle are both red
  change colors for grandfather and X
- Case 1 double red conflict
  - Uncle is black
- Reconstruction



11.6.4 Algorithm for Deletion

- First call the deletion algorithm for BST
  - if the node to be deleted has one or more external null pointers, then delete it directly.
  - otherwise, find its successor in its right subtree and exchange values (not colors), and then delete it.
- Let v be the internal node to be deleted, w is the external node to be deleted, X is the brother of w.
  - if either v or X is red, then color X black.
  - otherwise, X needs to be labelled double black, and reconstruct the tree according to its brother C.



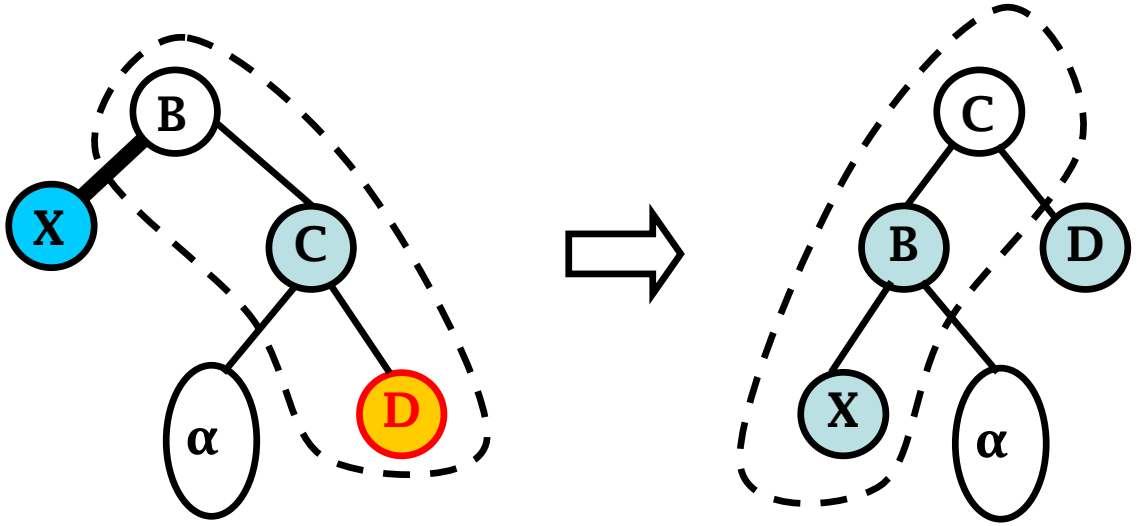
Adjustment according to C, the brother of a double blacked node X

Suppose X is a left child (the other case is symmetric)

- **Case 1:** C is black, and has a red child
  - reconstruct and thus end deleting.
- **Case 2:** C is black, and has two black children
  - exchange colors
  - if its father node B is originally black, then further adjustment might be needed from B upwards.
- **Case 3:** C is red
  - exchange nodes
  - C is converted to its father, and continue deleting according to case1 or case 2

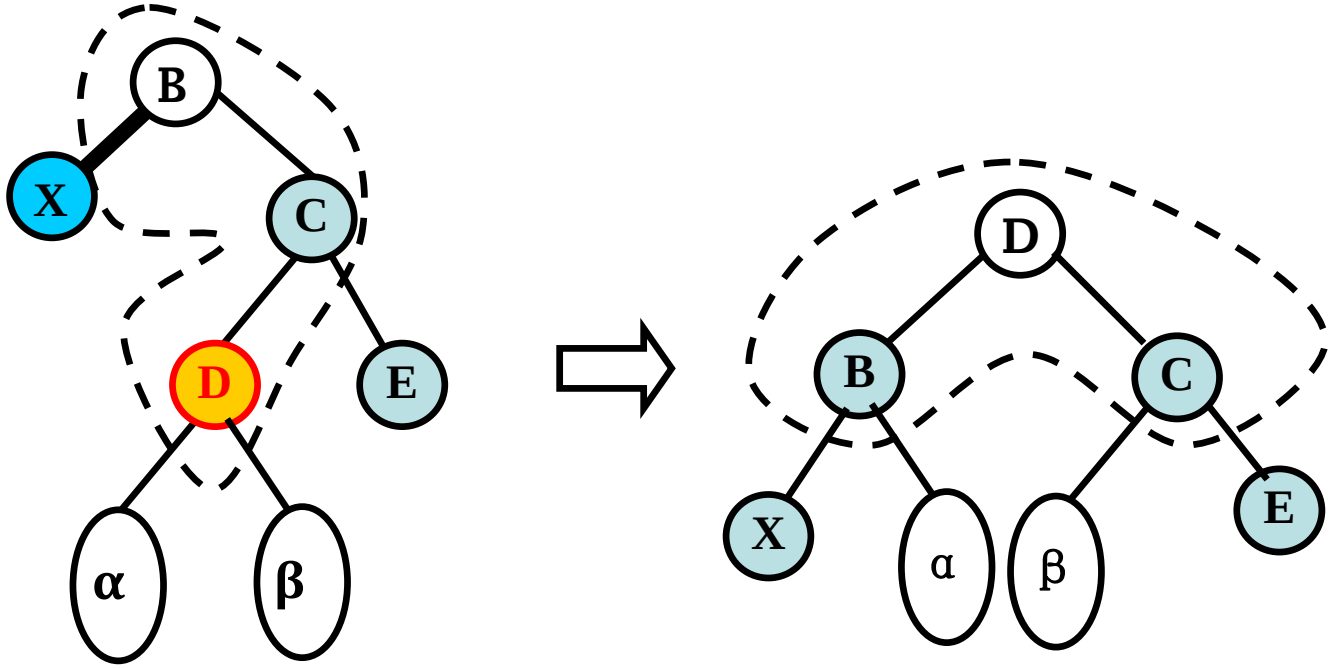
Case 1(a) reconstruction: red nephew on the other side

- Raise its brother C
- C gets the color of its original father node
- Color B and C black, leaving the others unchanged



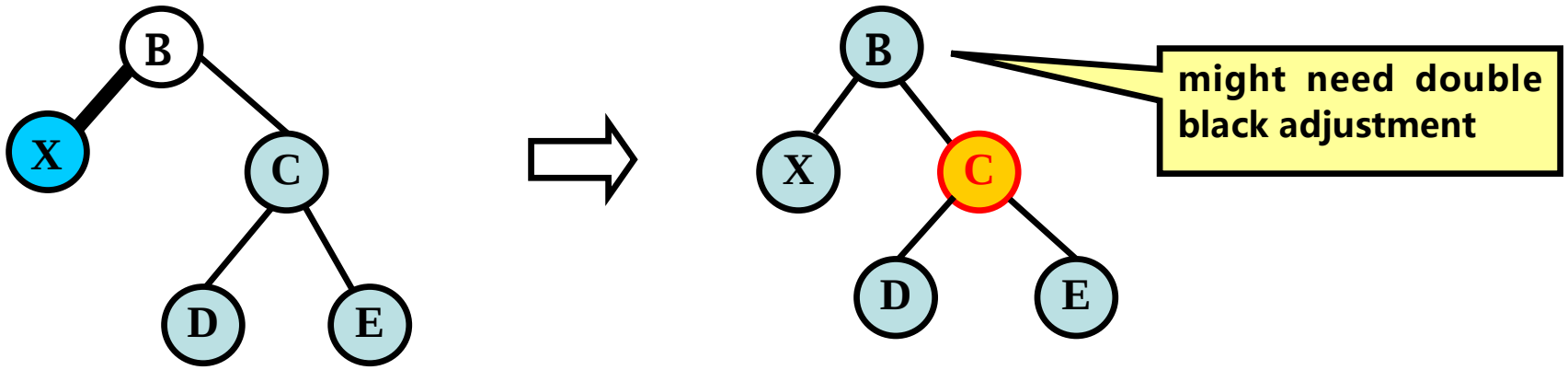
Case 1(b) reconstruction: red nephew on the same side

- Raise D to the father of C, and D gets the original color of B, B is colored black



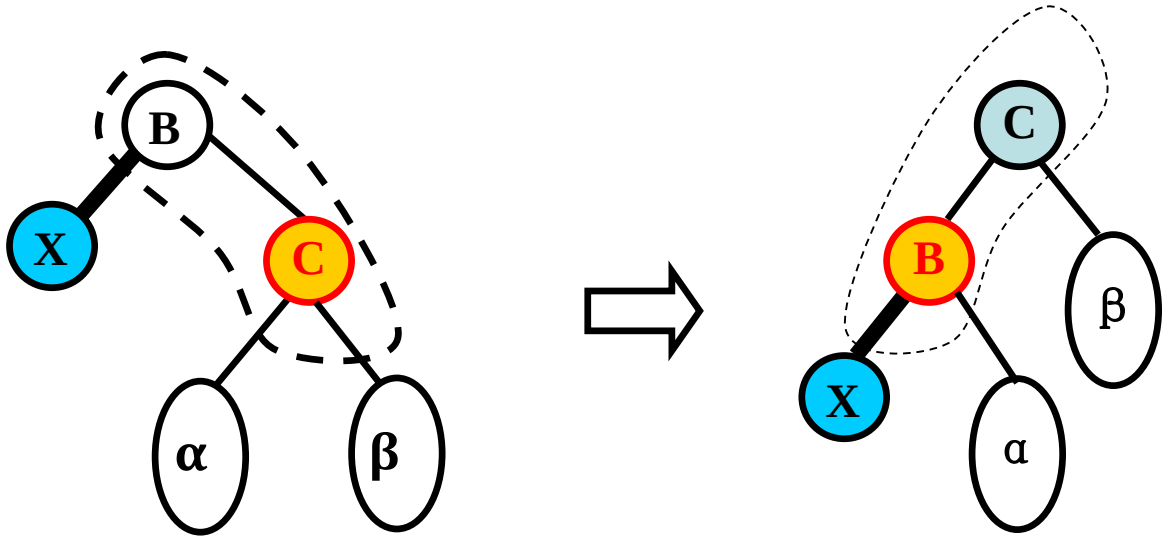
Case 2: brother is black, and have two black children

- Color C red, and color B black.
- If B is originally red, then the algorithm is completed.
- Else, perform double black adjustment on B.



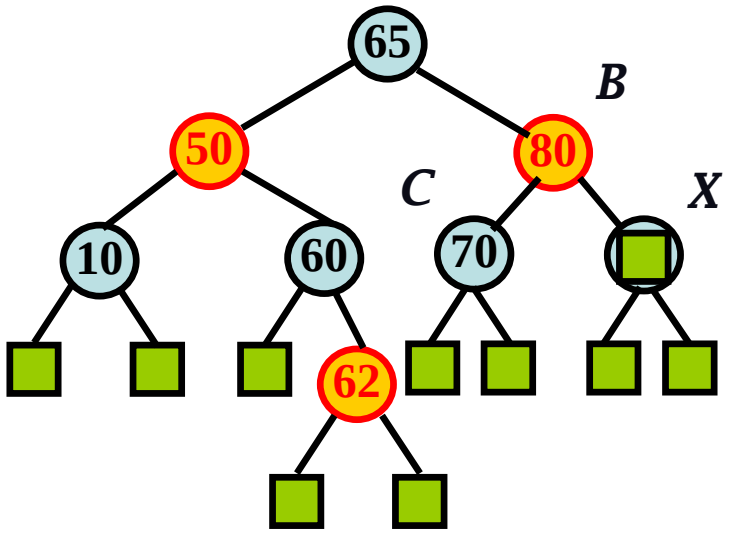
Case 3: brother C is red

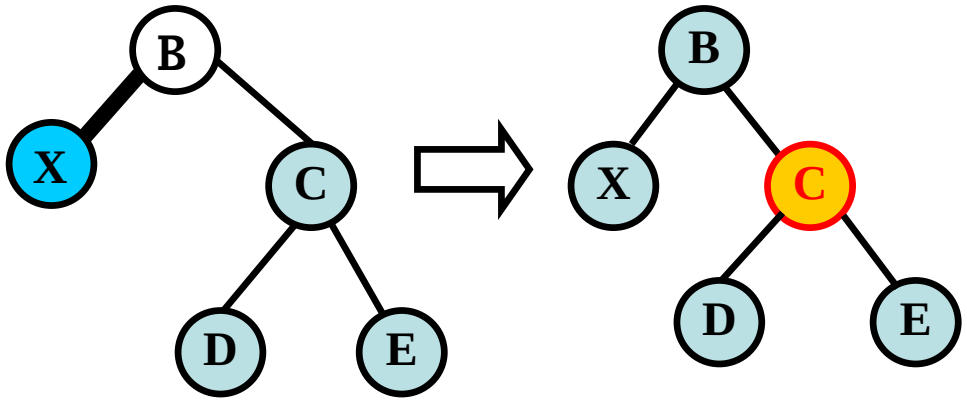
- Rotation
- X is still the double blacked node, and thus the case converts to one of the two cases before



Delete 90

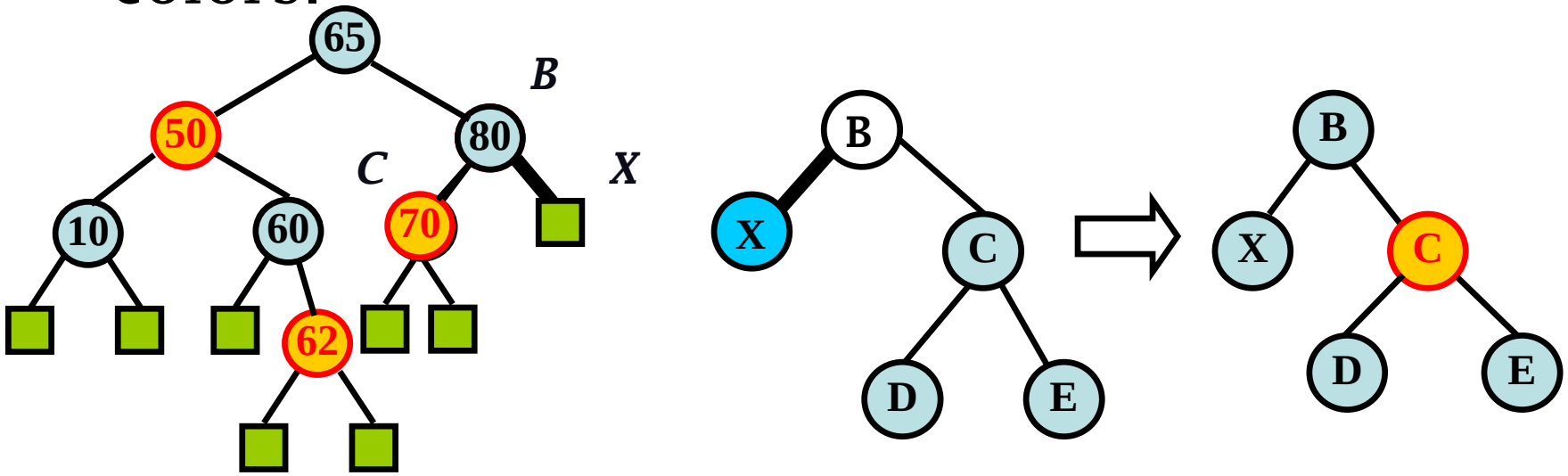
- Change the current node to the right black leaf of 80.
- C is black and has two black nodes: case 2





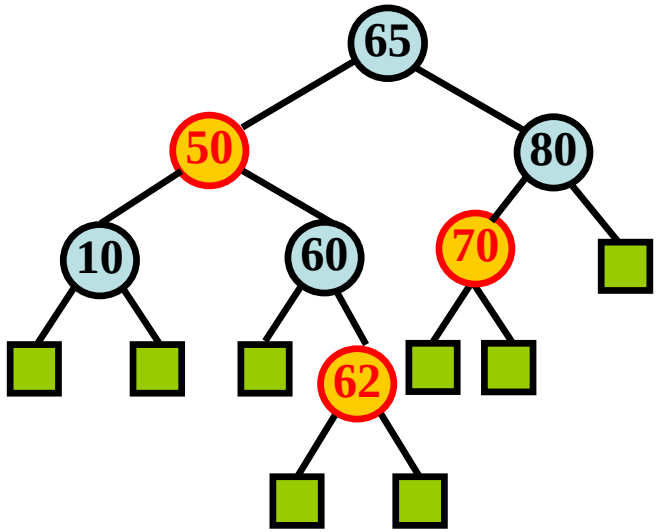
Delete 90

- Change the current node to the right black leaf child of 80.
- C is black and has two black nodes: case 2, exchange colors.



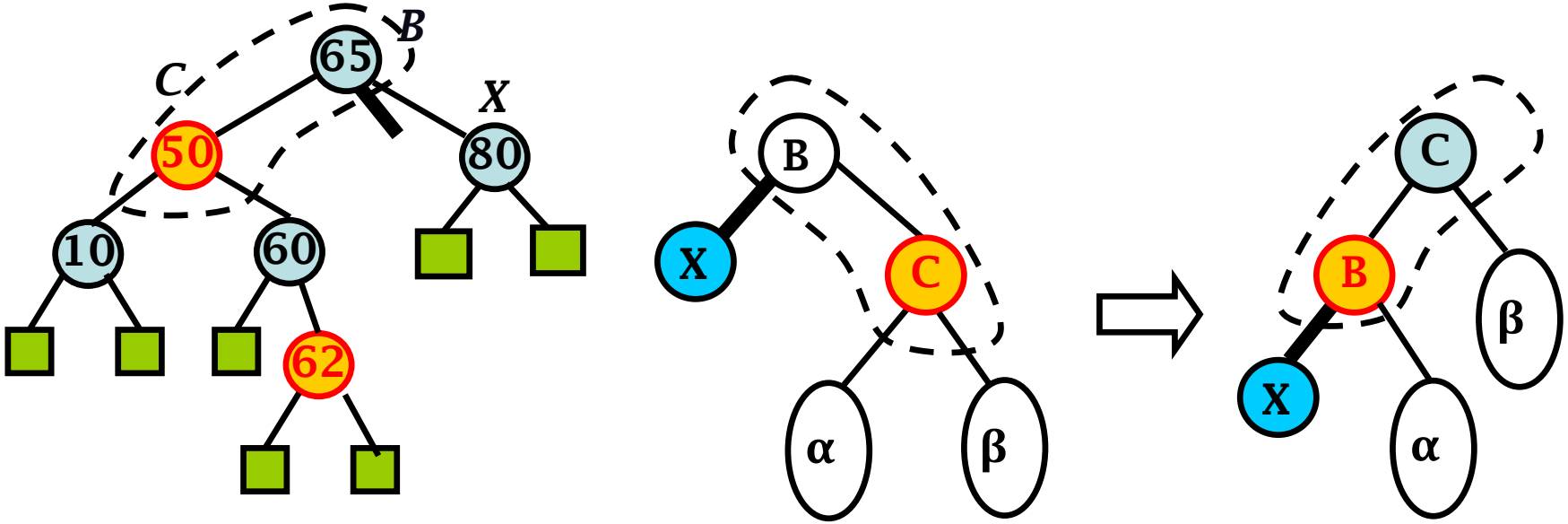
Delete 70

- Red node, no need to adjust.



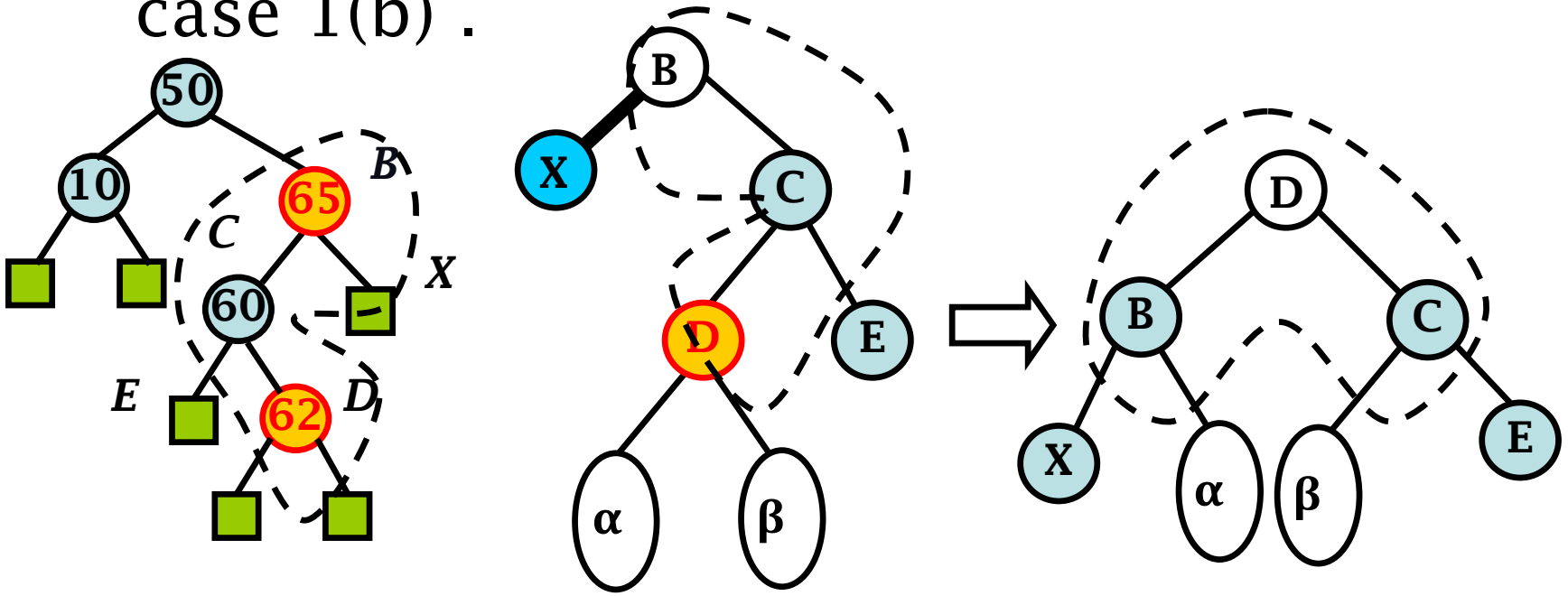
Delete 80

- Change the current node X to the right black leaf child of 65.
- C is red, exchange nodes according to case 3.



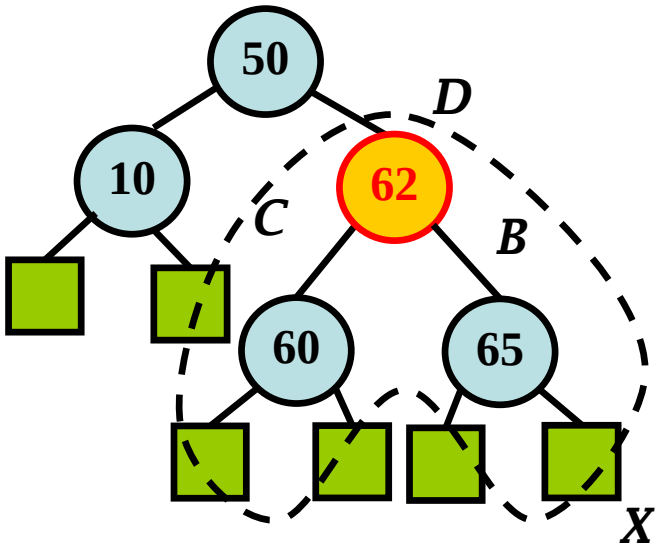
Delete 80 (adjustment)

- C is black, with black left child and red right child: reconstruction according to case 1(b) .



Delete 80

- Adjustment completed.



Time Cost of Deletion

- Average and worst case $O(\log_2 n)$
  - Bottom-up adjustment.
- Construction of RB-tree
  - (data, left pointer, right pointer, color, father pointer)
- Top-down recursive insertion/deletion adjustment
  - (data, left pointer, right pointer, color)
  - must keep records of tracks if the process is not recursive

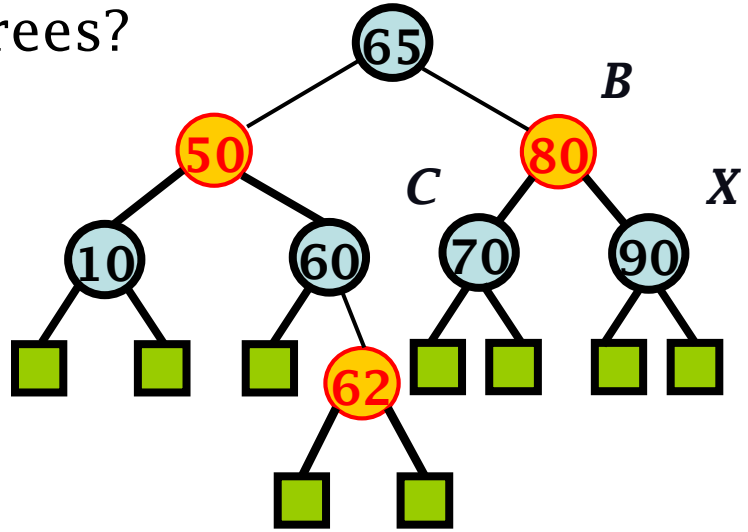
From Red-Black Tree to 2-3-4 Tree

- Suppose each red node of a RB-Tree is absorbed to its black father, and let its children be the children of its black father (ignore the change of key values).

- When the red children of a black node are all absorbed, what are its possible degrees?
  - 2, 3, 4

    Thus became a 2-3-4 tree.

    (A B-tree in order 4)
- What about the height of its leaves?
  - Leaves are on the same level.



Data Structures and Algorithms

Thanks

the National Elaborate Course (Only available for IPs in China)
http://www.jpk.pku.edu.cn/pkujpk/course/sjjg/
Ming Zhang, Tengjiao Wang and Haiyan Zhao
Higher Education Press, 2008.6 (awarded as the "Eleventh Five-Year" national planning textbook)