Data Structures

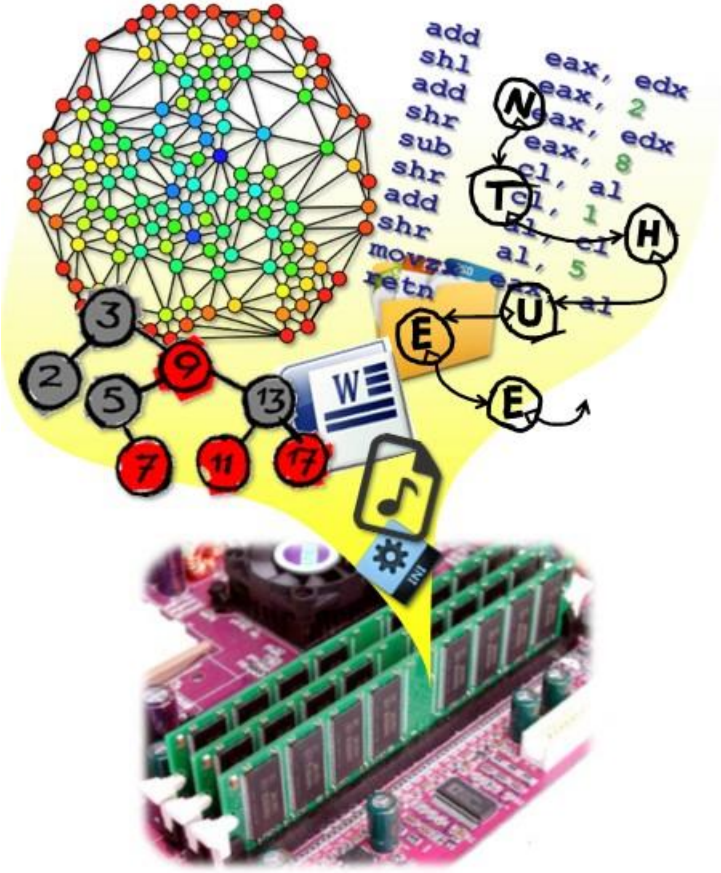
Selected Topics

Red-Black Trees

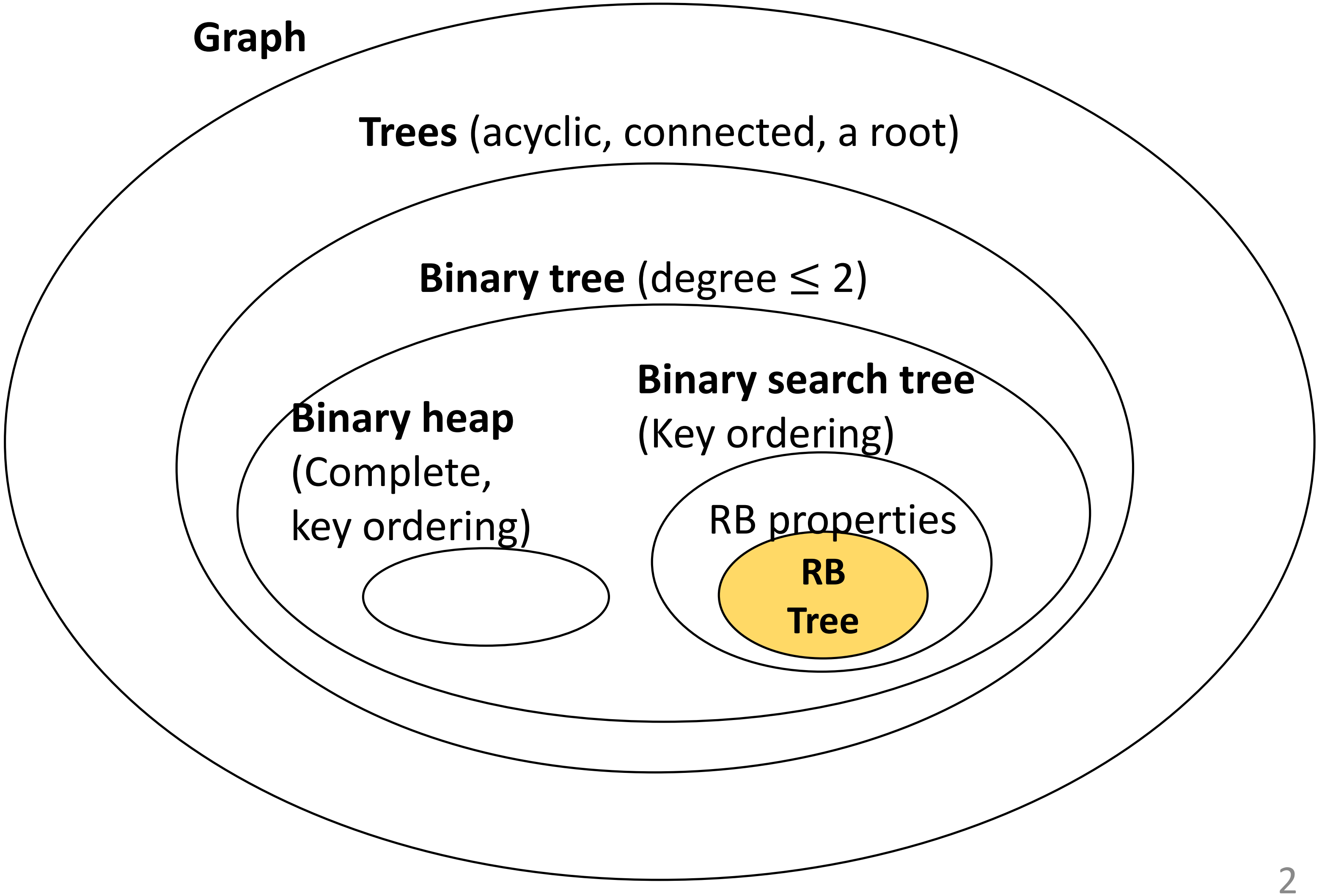
Prof. Ren-Shuo Liu

NTHU EE

Spring 2018

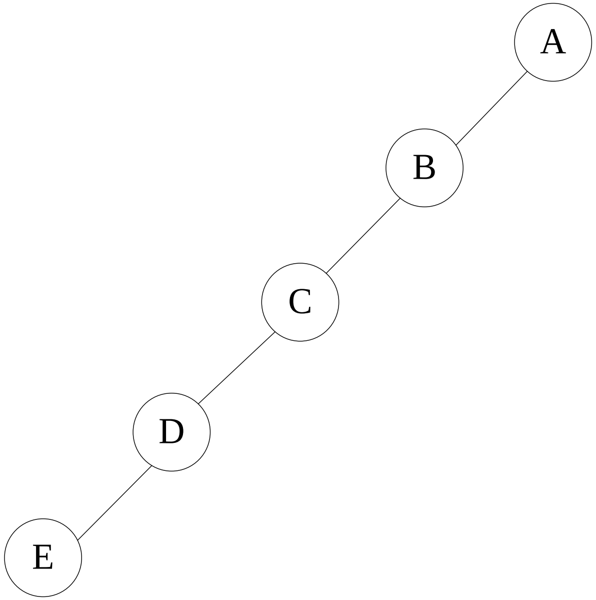


RB Trees vs Other Data Structures



Drawback of Standard BST

- In some worst cases, a tree is skewed, and a tree operation such as searching becomes O(n) time

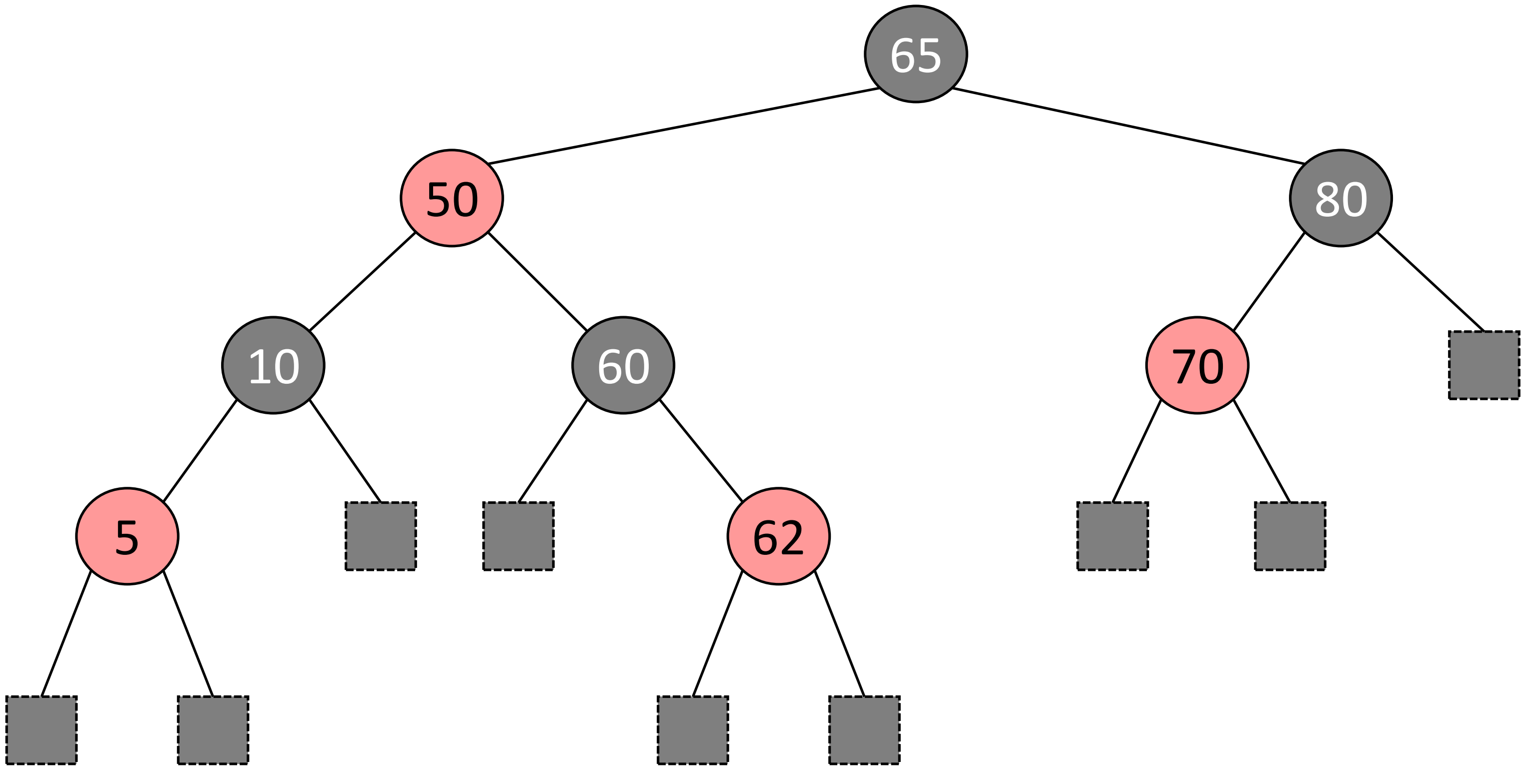


Red-Black Trees (RB Trees)

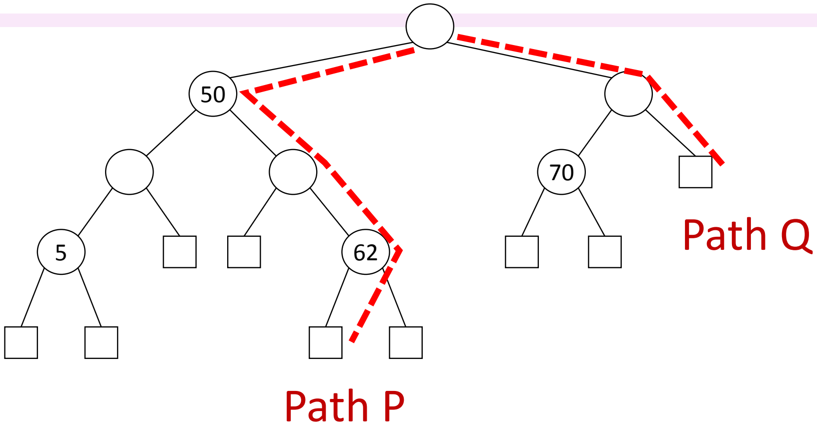
- RB tree is a binary search tree
  - Every node has extra 1-bit information denoting whether the node is colored red or black
  - Empty subtrees are viewed as nil nodes
  - Satisfies the following red-black properties
    - Root node is black
    - Nil nodes are viewed as black
    - All root-to-nil paths have the same number of black nodes
    - No consecutive red nodes in a path

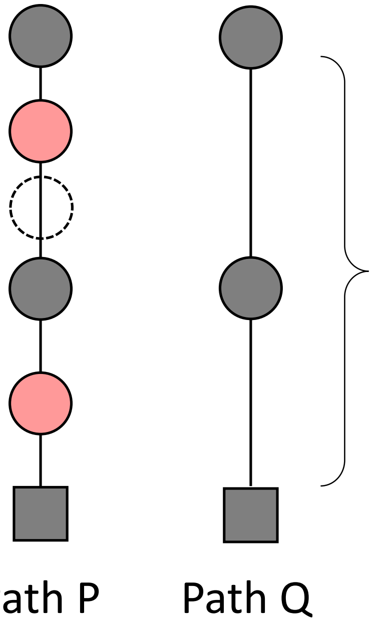
- These properties guarantee balanced binary search trees

RB Tree Example



So-Called Balanced Trees





4. no consecutive red nodes

1. Root is black

3. Same number of black nodes

2. Nil node is black

From 1, 2, 3, and 4
→ Root-to-nil path lengths in an RB tree vary no more than 2x
→ Tree is balanced

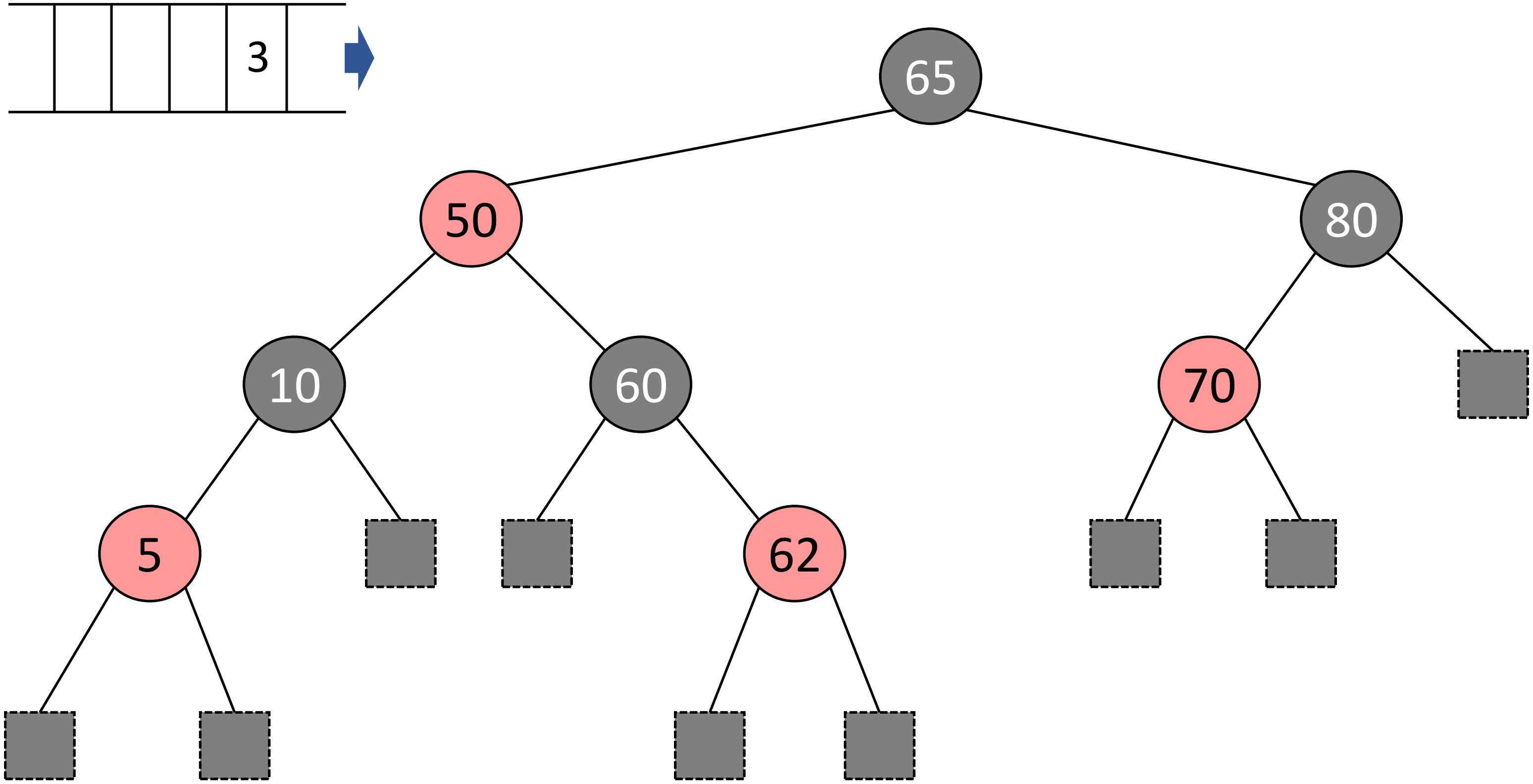
Red-Black Tree Operations

- Search (the same as a standard BST)
- Insertion
- Deletion

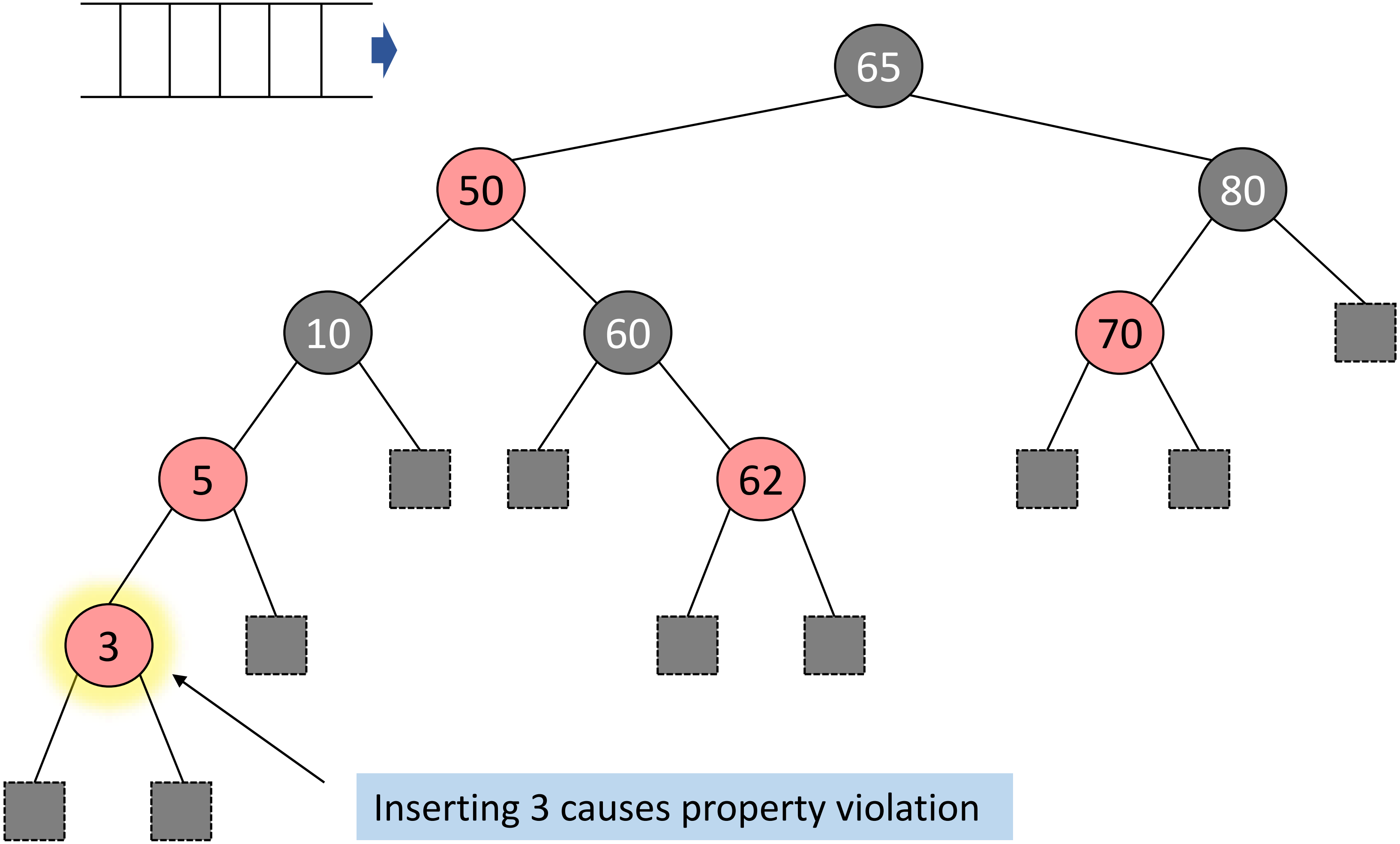
Insertion

- If the tree is non-empty
  - Perform standard BST insertion
  - Make the new node red
  - Check the color of the parent of the new node
    - Black
      - Insertion is done
    - Red
      - Two consecutive red nodes appear
      - Tree is imbalance
      - Need rebalancing
- If the tree is empty
  - The new node becomes the root
  - Make the new node black
  - Insertion is done

RB Tree Insertion Example

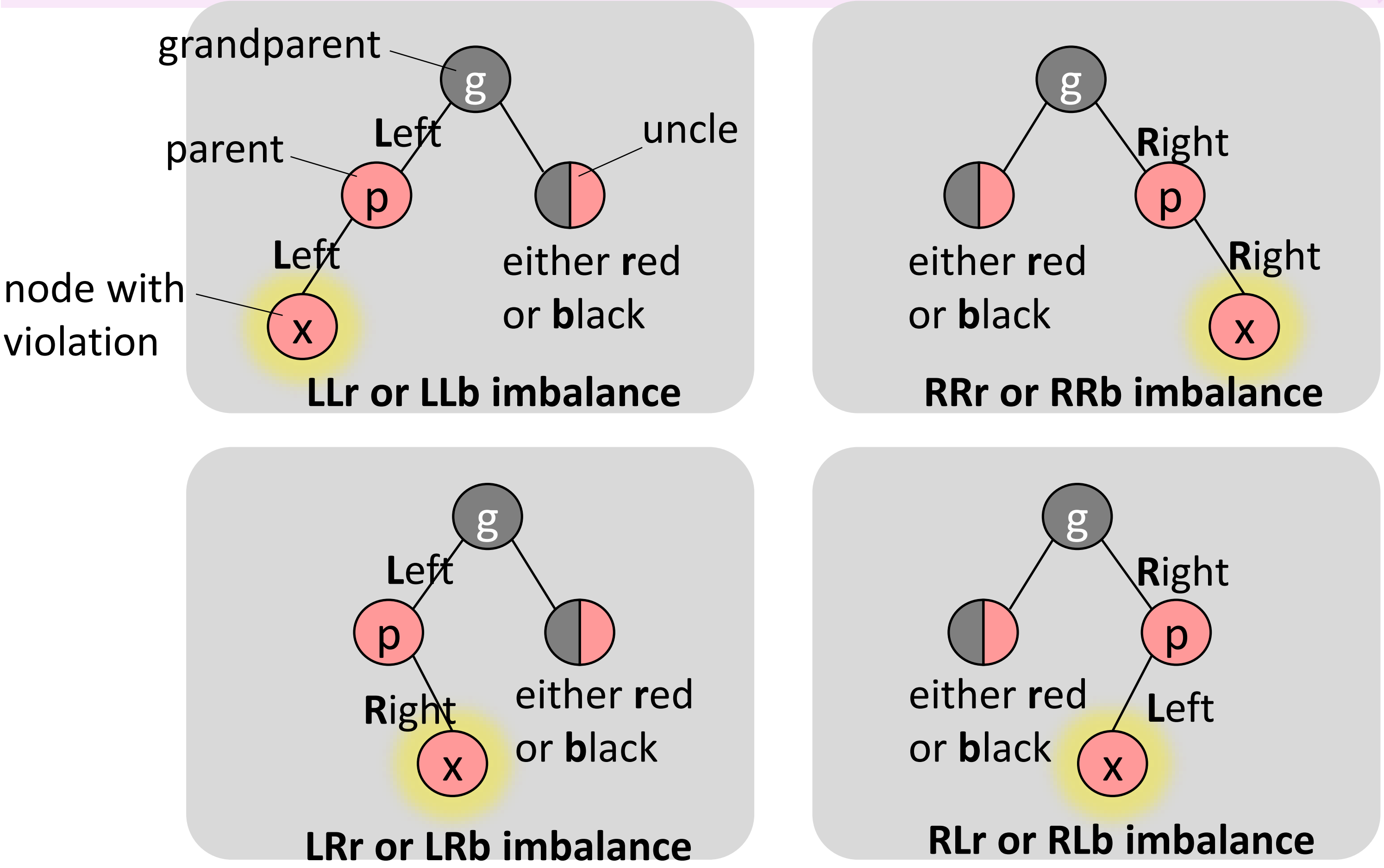


RB Tree Insertion Example



Inserting 3 causes property violation

Violation (Imbalance) Types

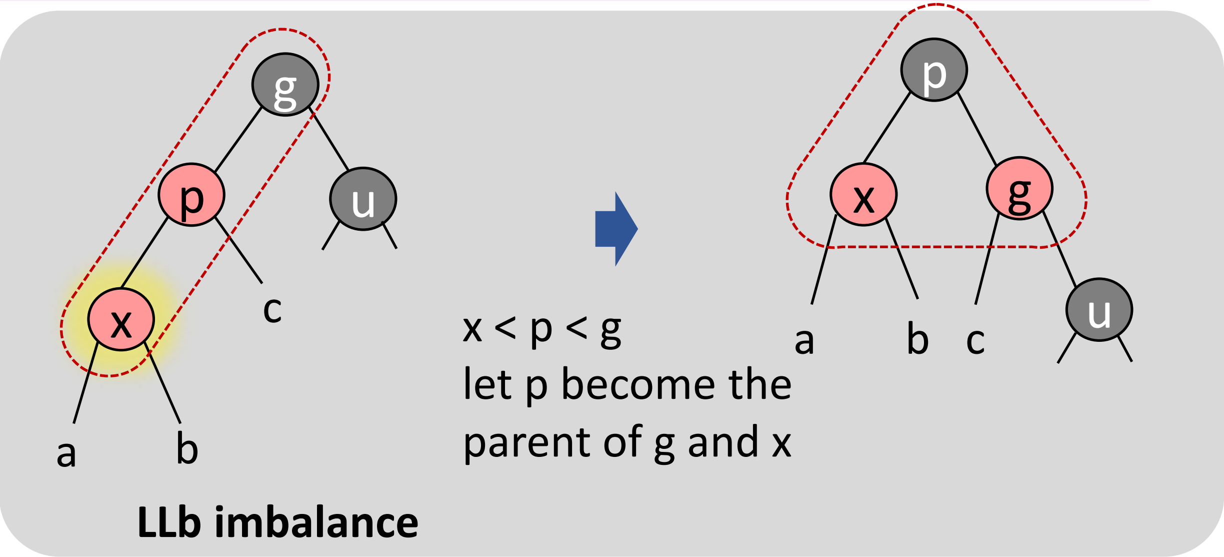


- Although there are eight possible cases, we only focus on whether the uncle is red or black

Black Uncle Case

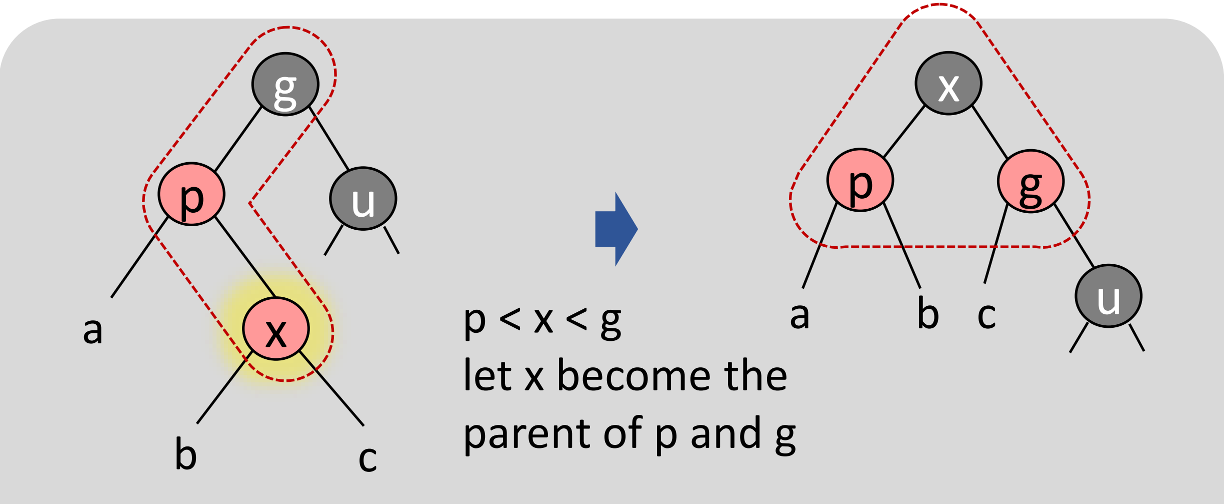
- Rotate to rebalance the number of red nodes
  - Move one red node to uncle's path

RRb and RLb are similar to LLb and LRb, respectively, and thus are omitted in the figure



x < p < g
let p become the parent of g and x

LLb imbalance

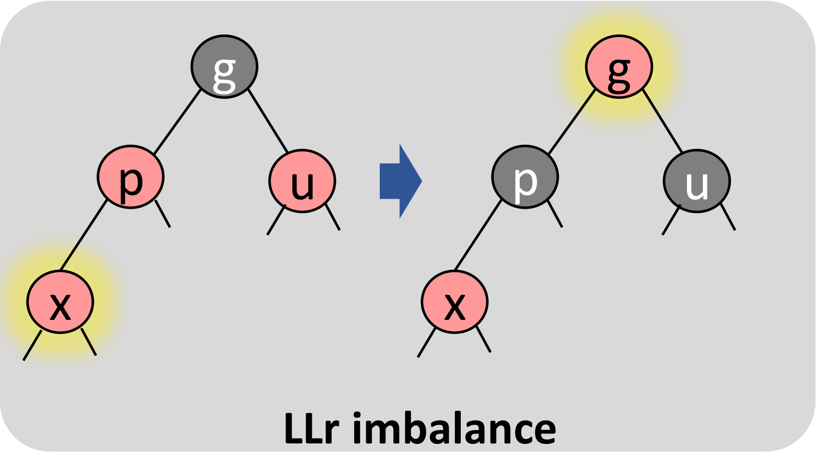


p < x < g
let x become the parent of p and g

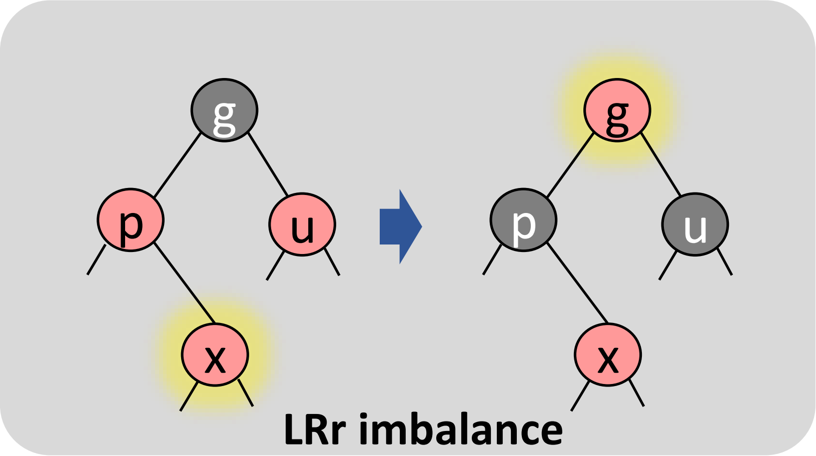
LRb imbalance

Red Uncle Case

- Perform color changes
- If the parent of the grandparent is red
  - Violation is propagated two levels up the tree
- Otherwise, insertion is done



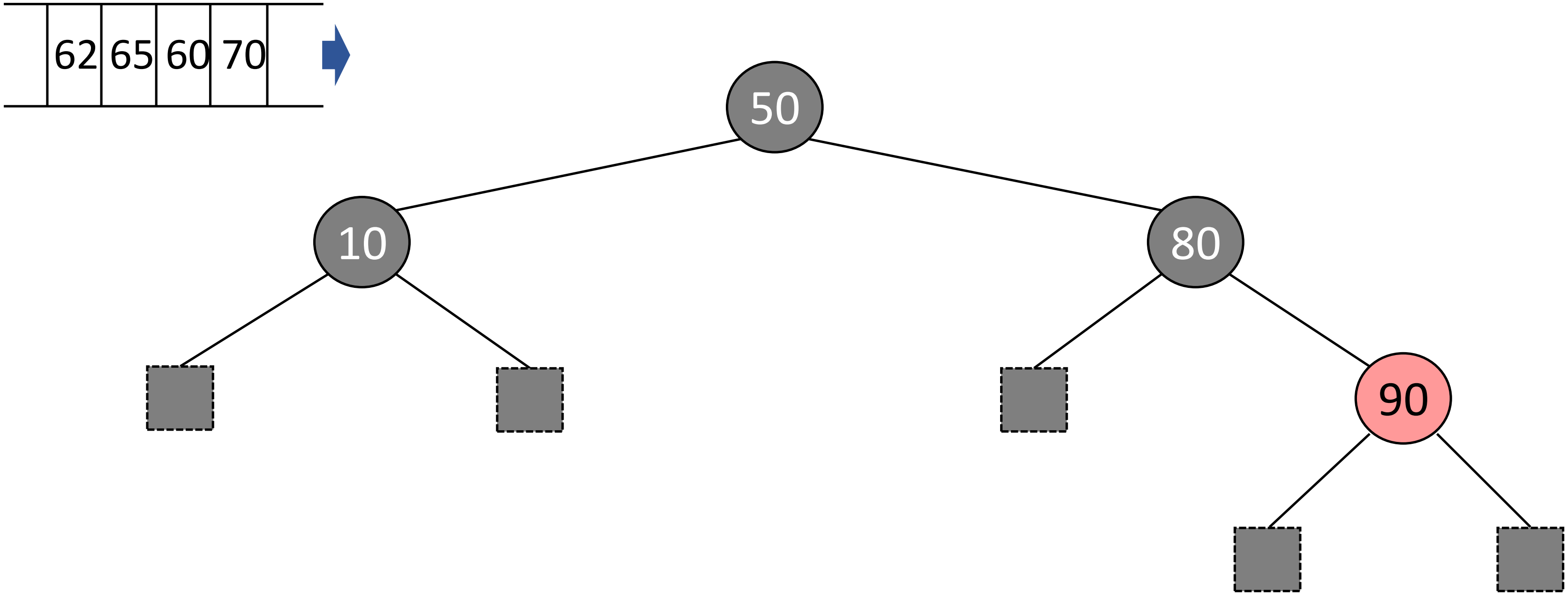
LLr imbalance



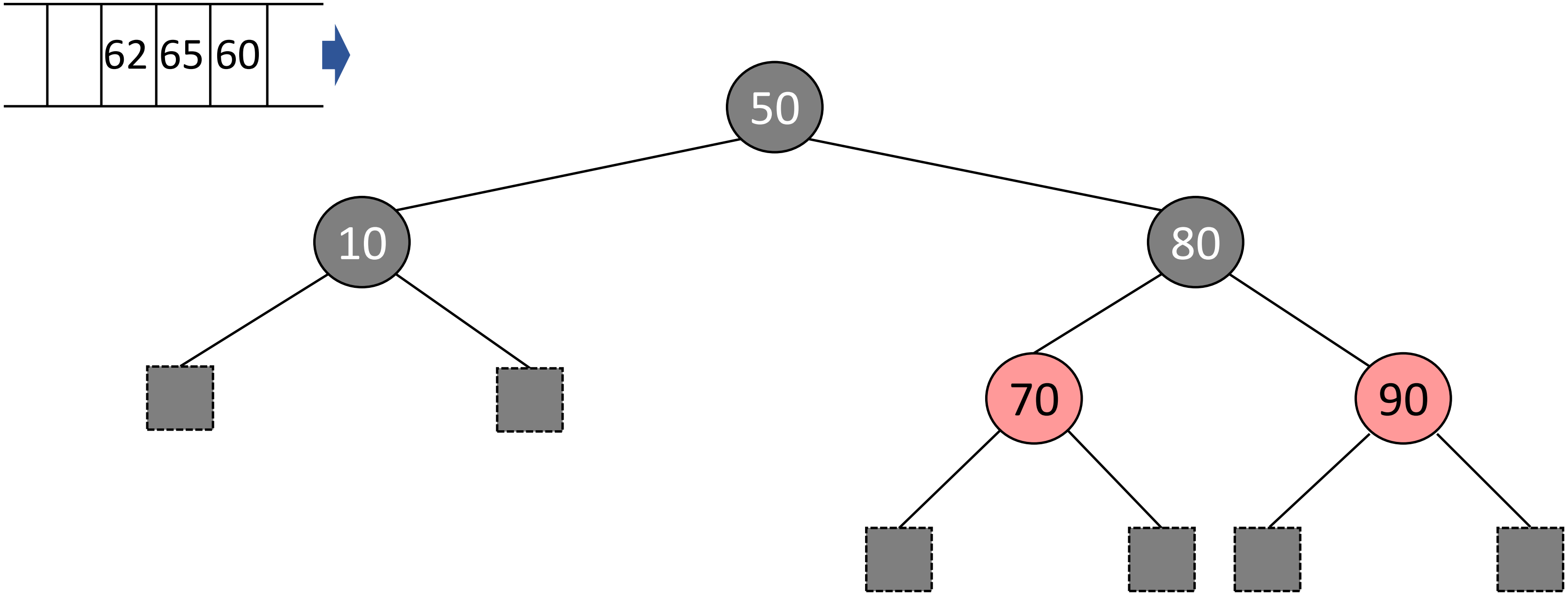
LRr imbalance

RRr and RLr are similar to LLr and LRr, respectively, and thus are omitted in the figure

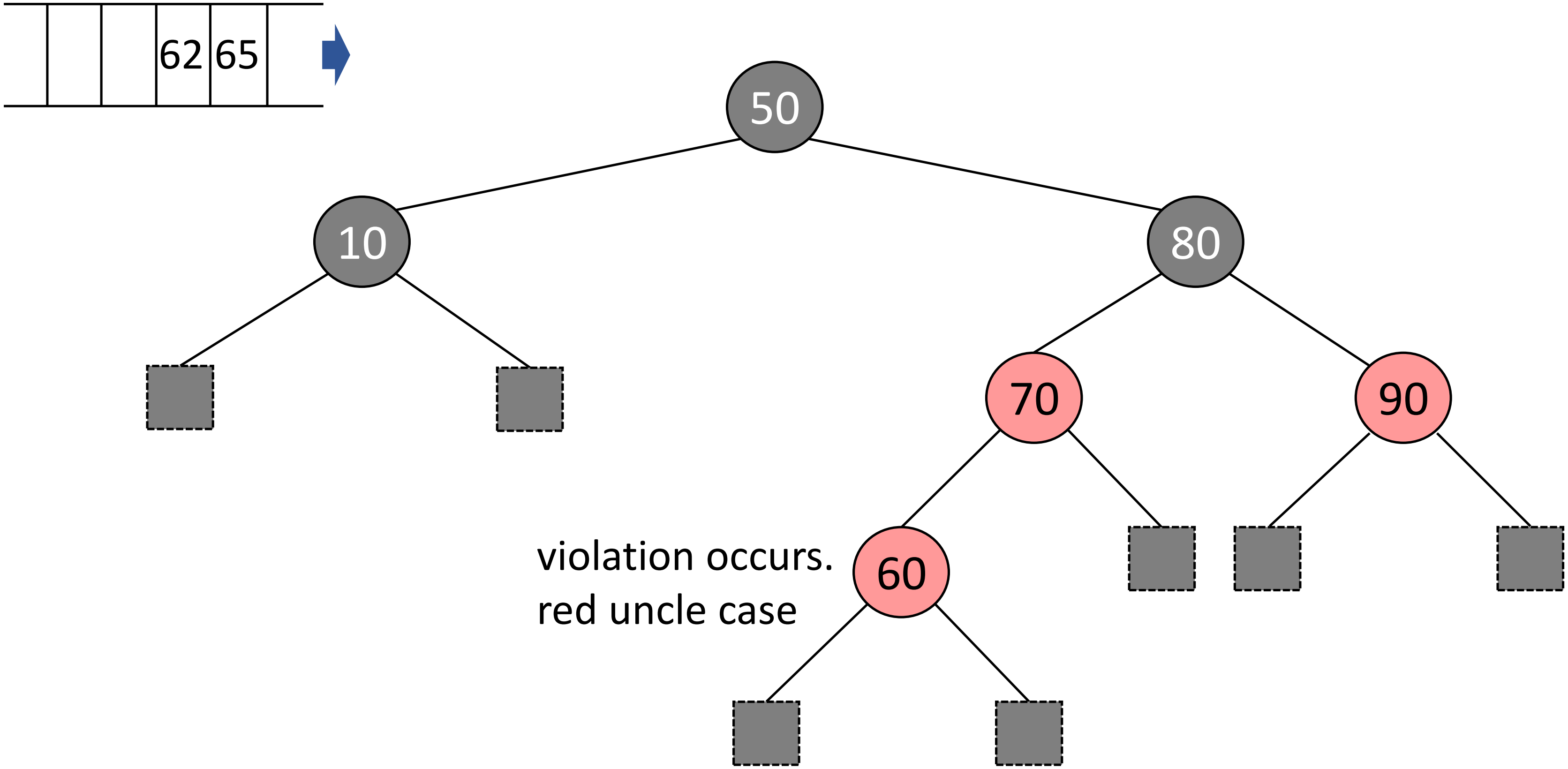
Example



Example

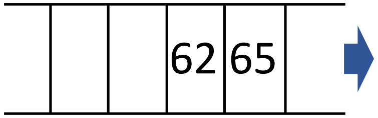


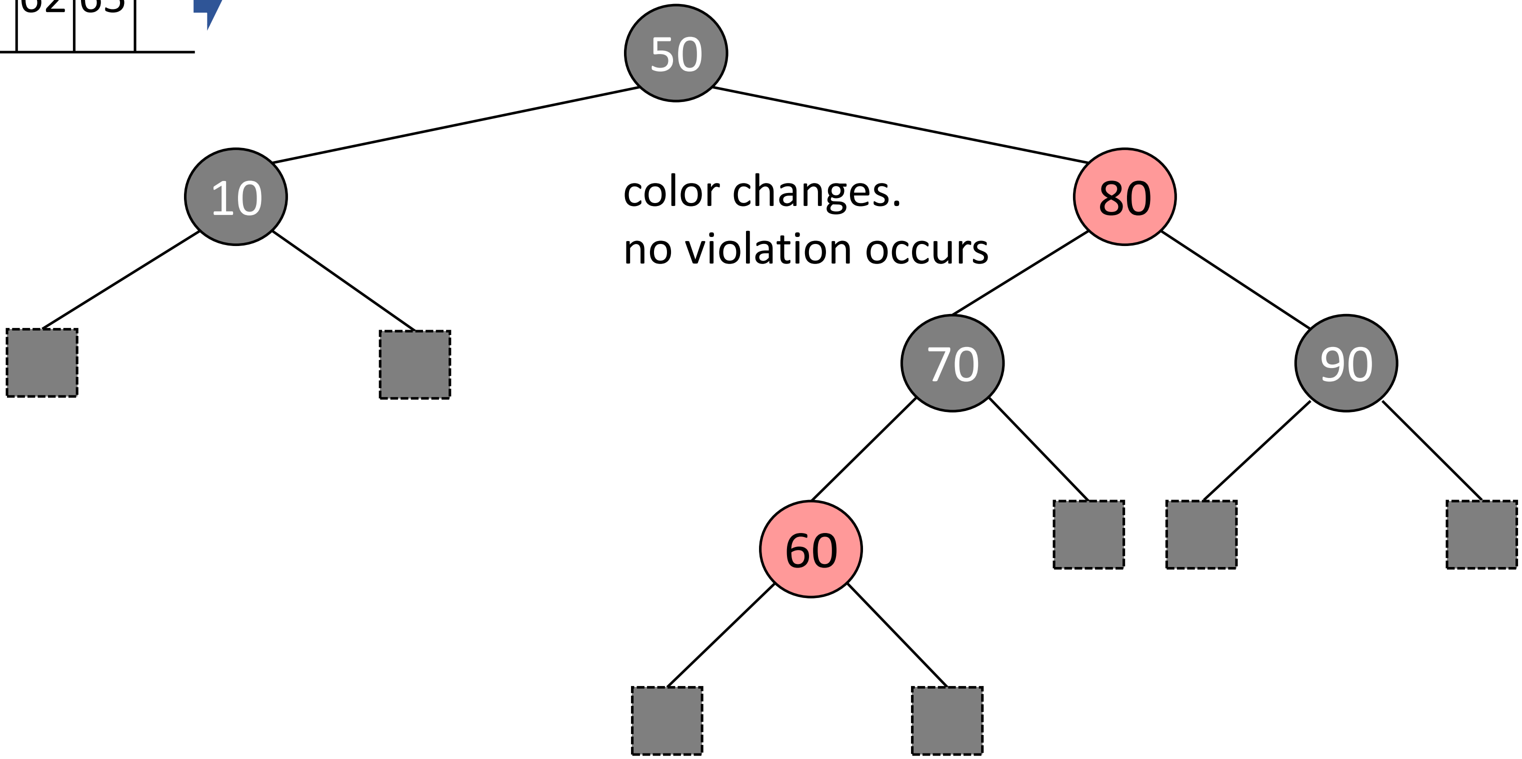
Example



violation occurs.
red uncle case

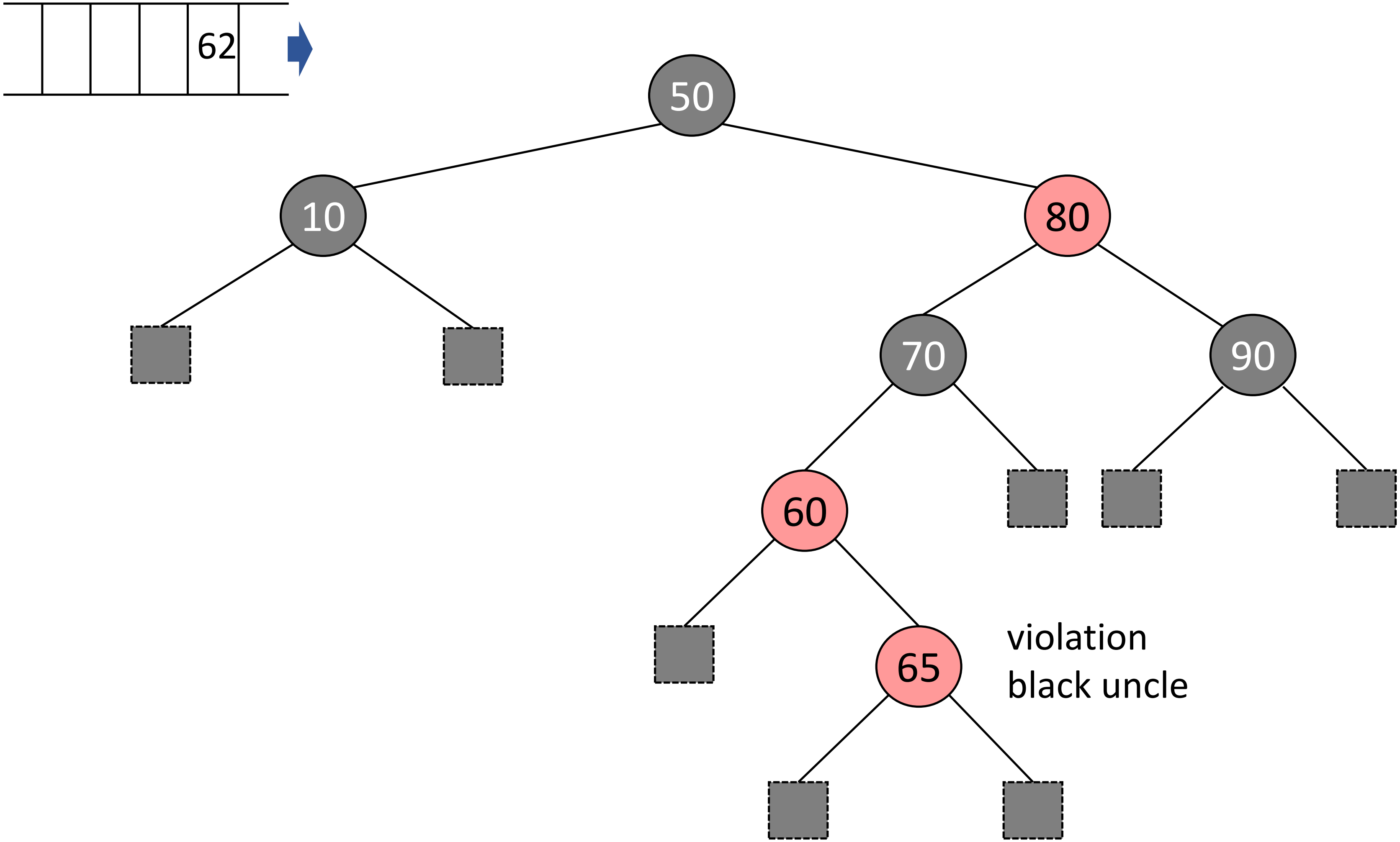
Example





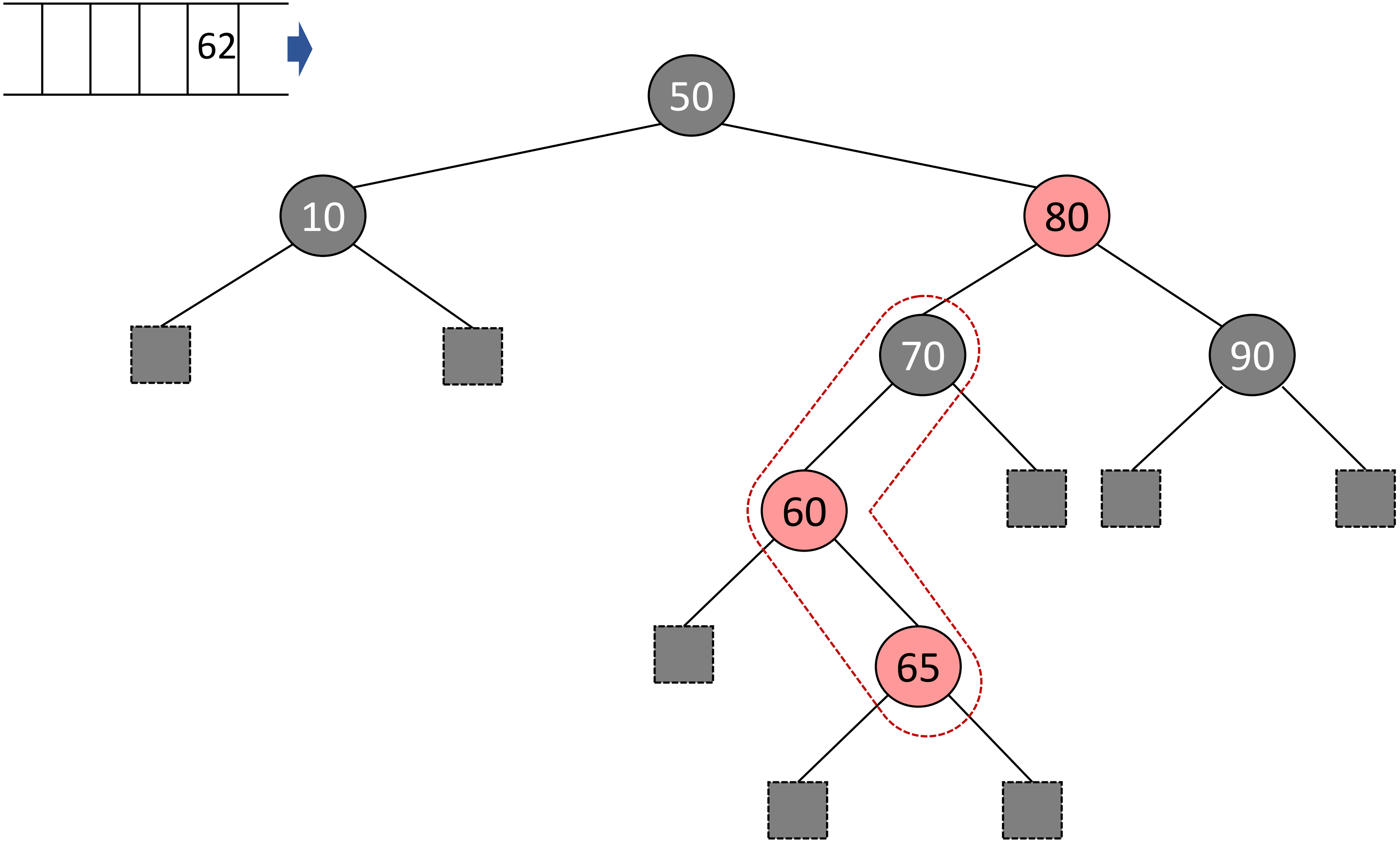
color changes.
no violation occurs

Example

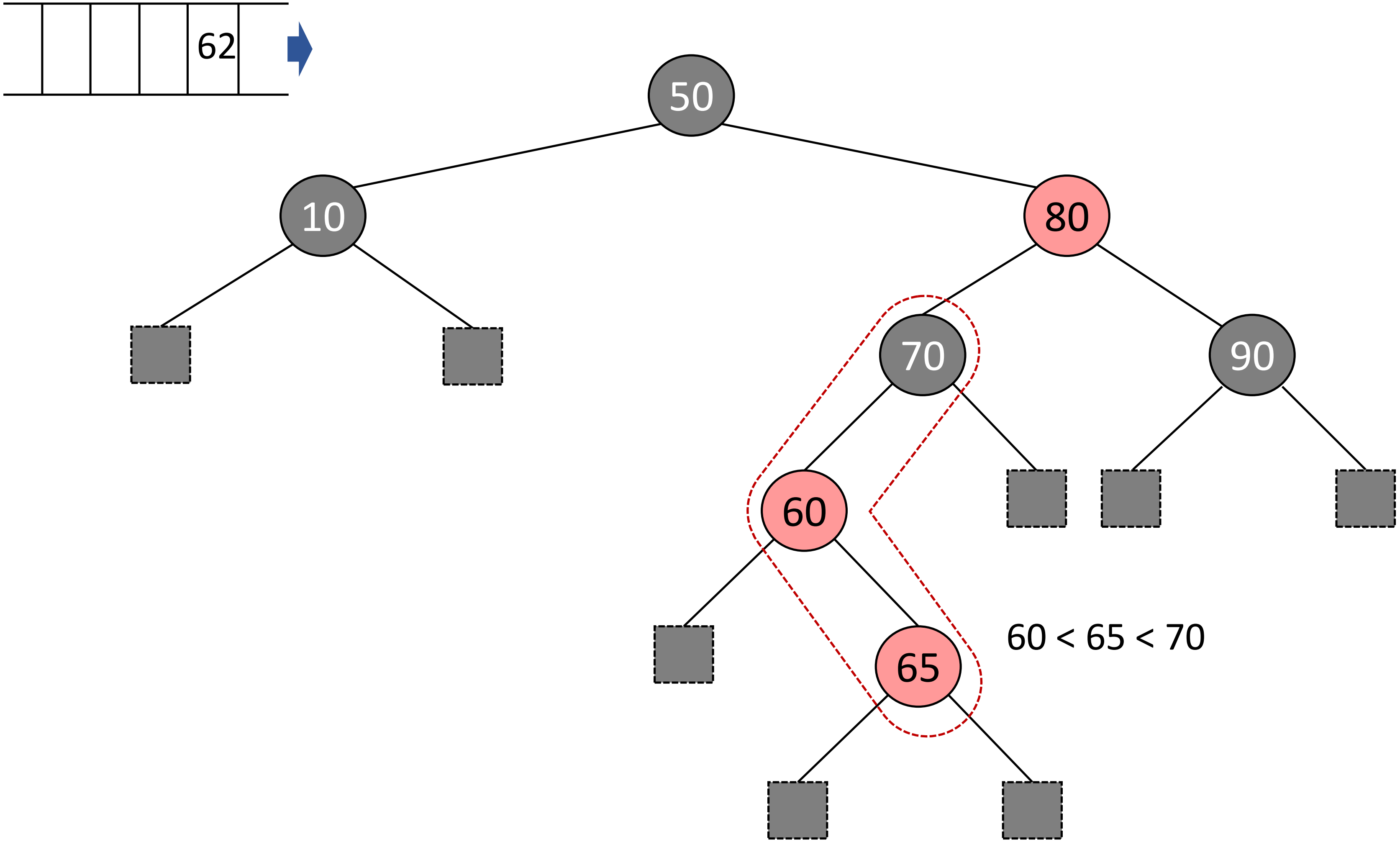


violation
black uncle

Example

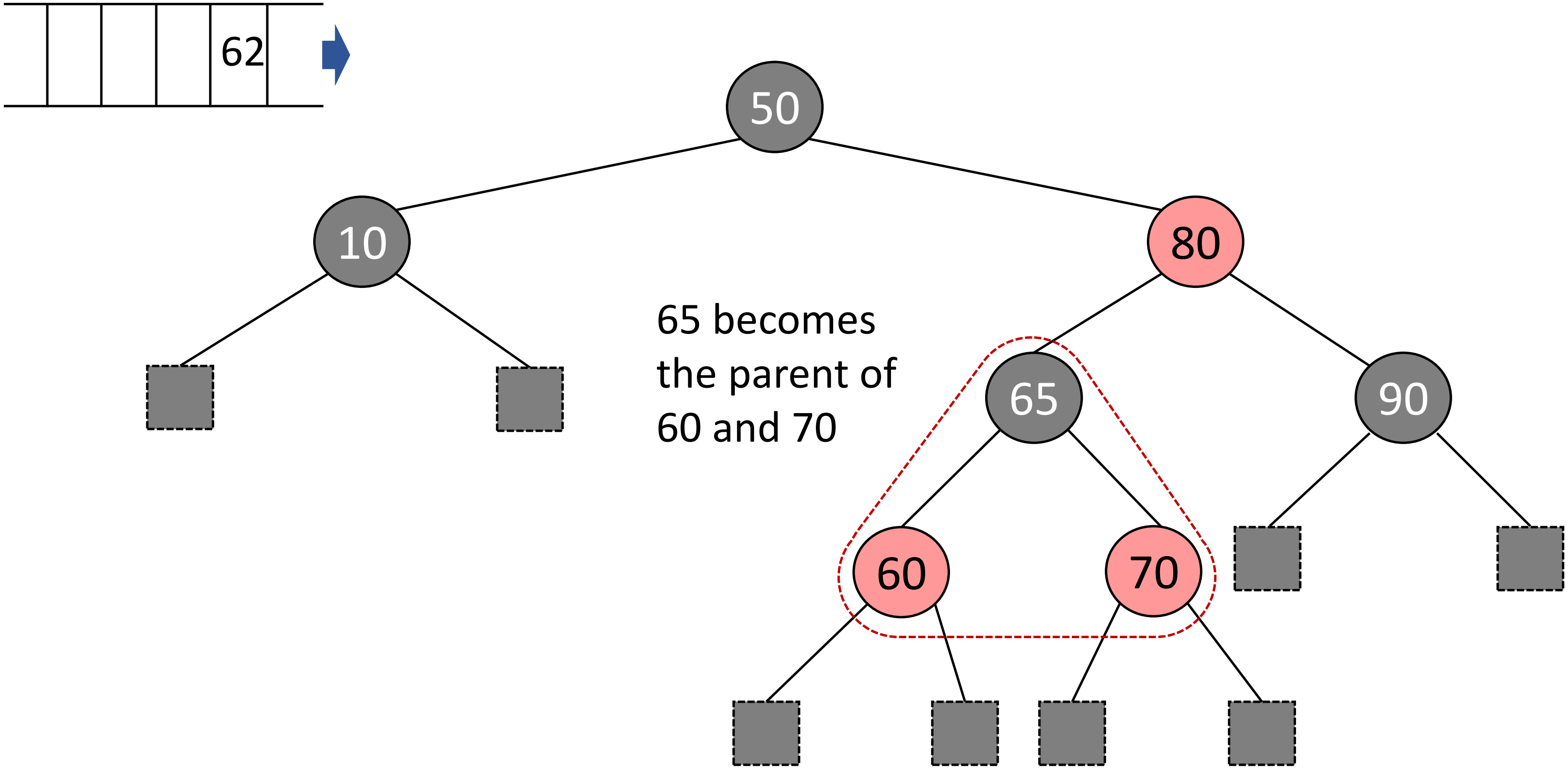


Example



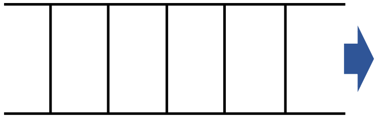
60 < 65 < 70

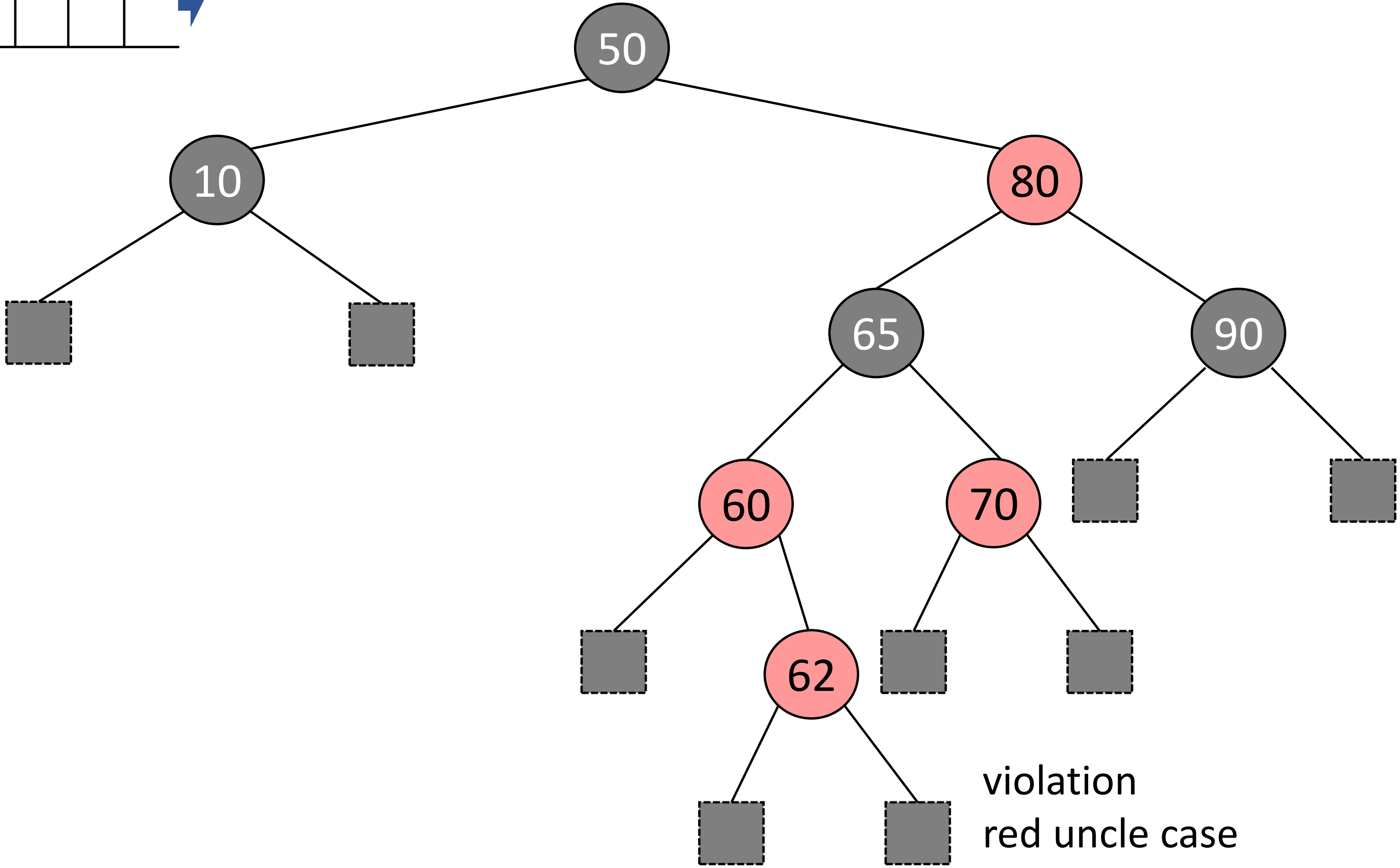
Example



65 becomes the parent of 60 and 70

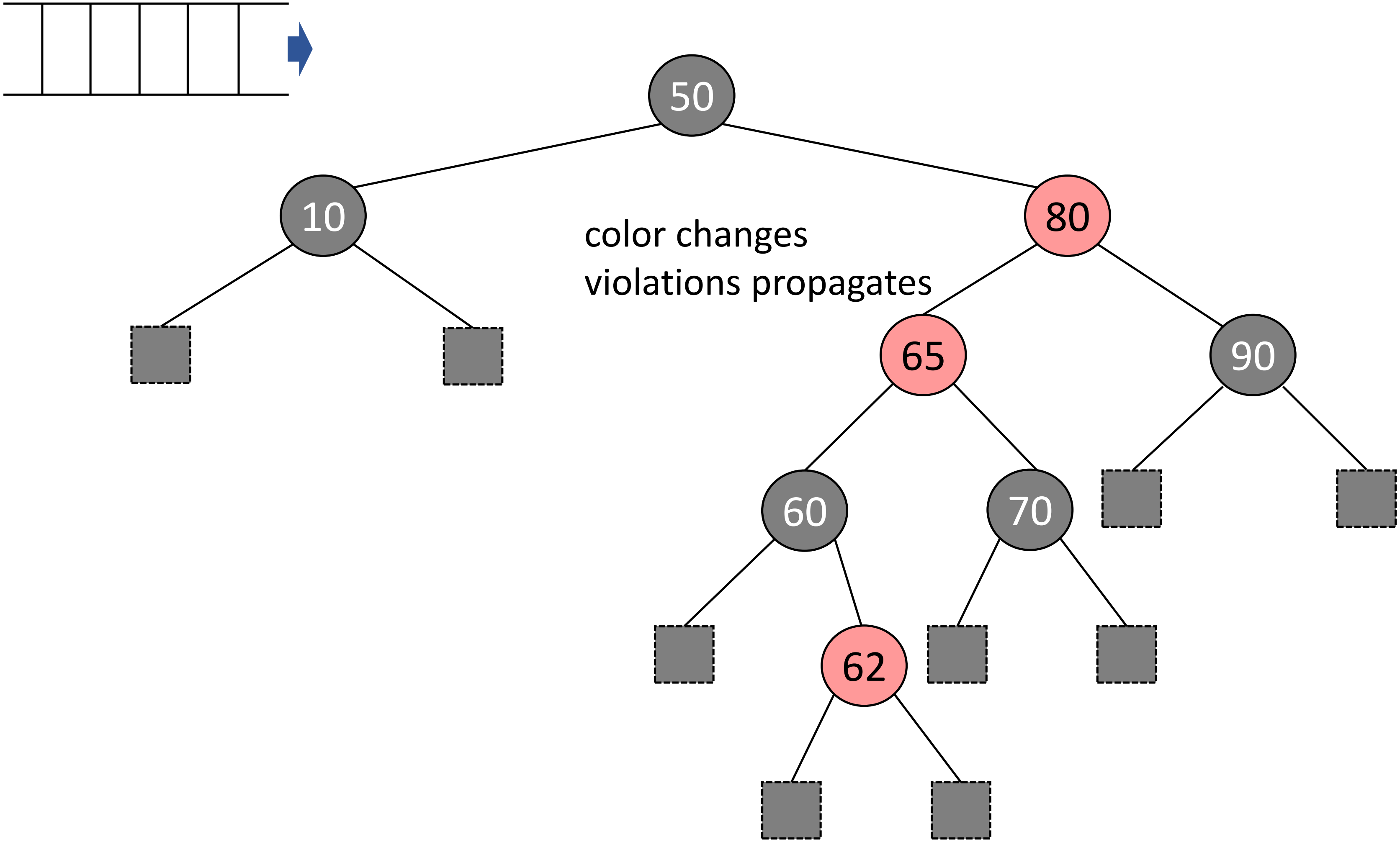
Example





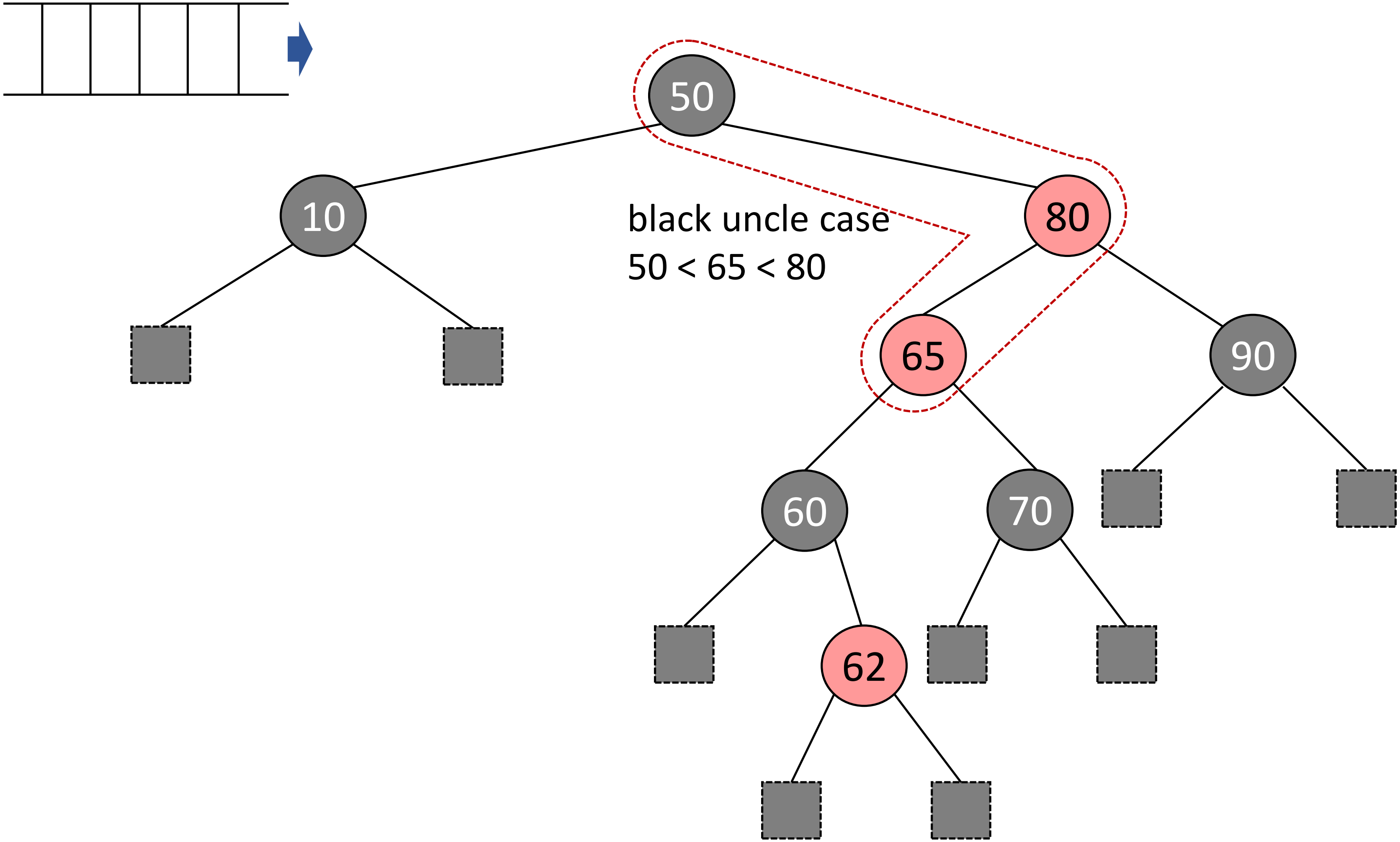
violation
red uncle case

Example



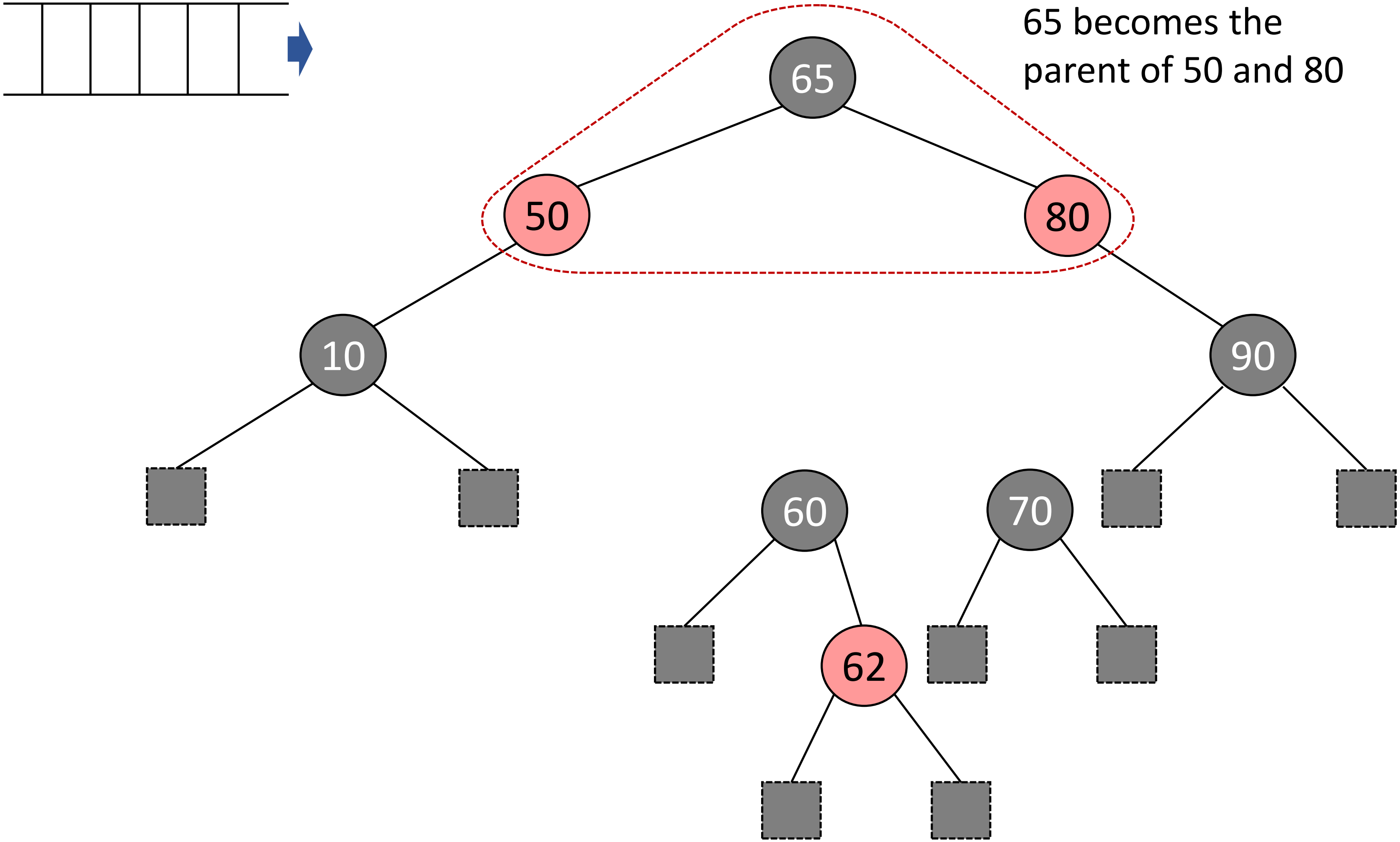
color changes
violations propagates

Example



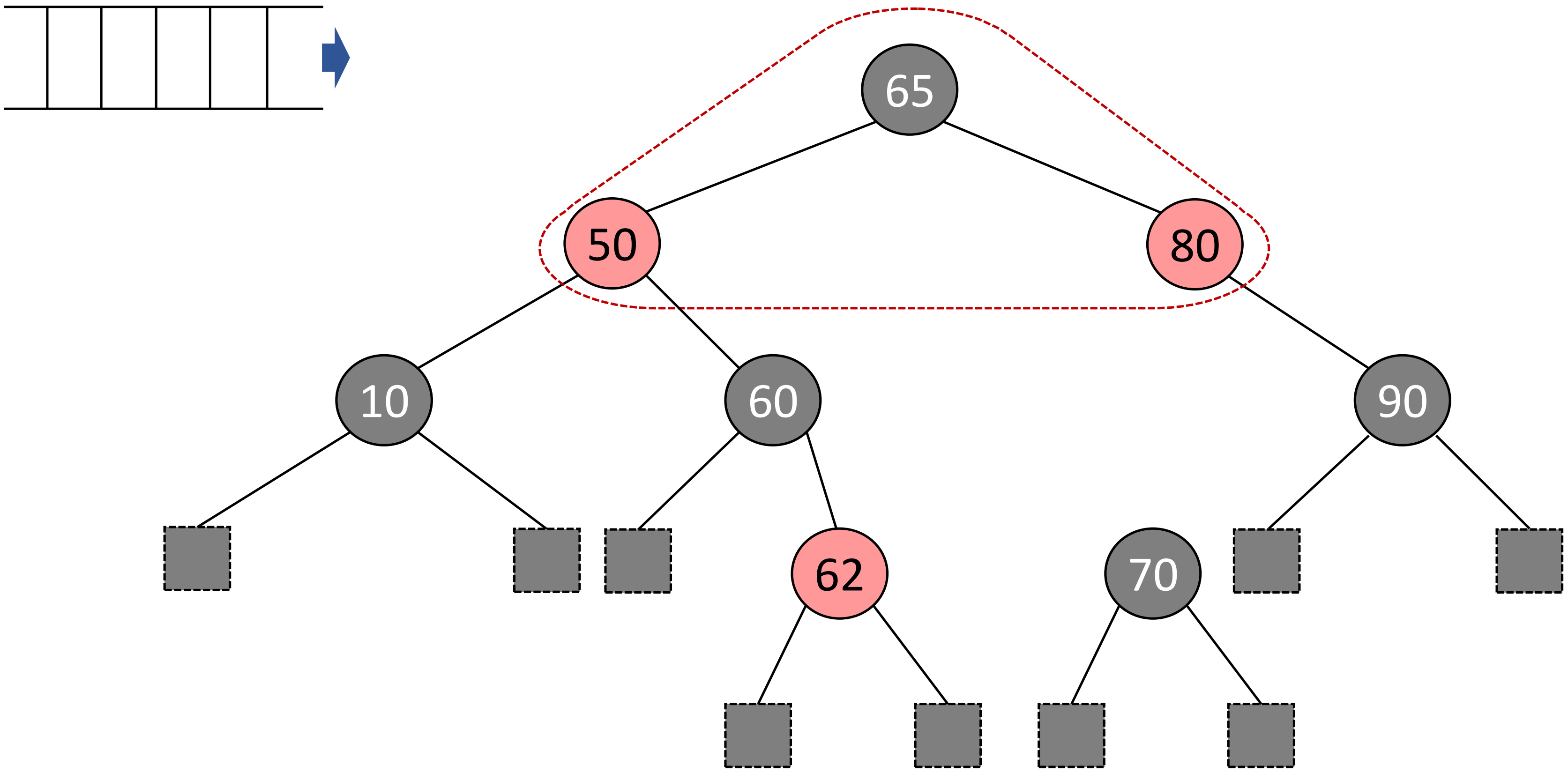
black uncle case
50 < 65 < 80

Example

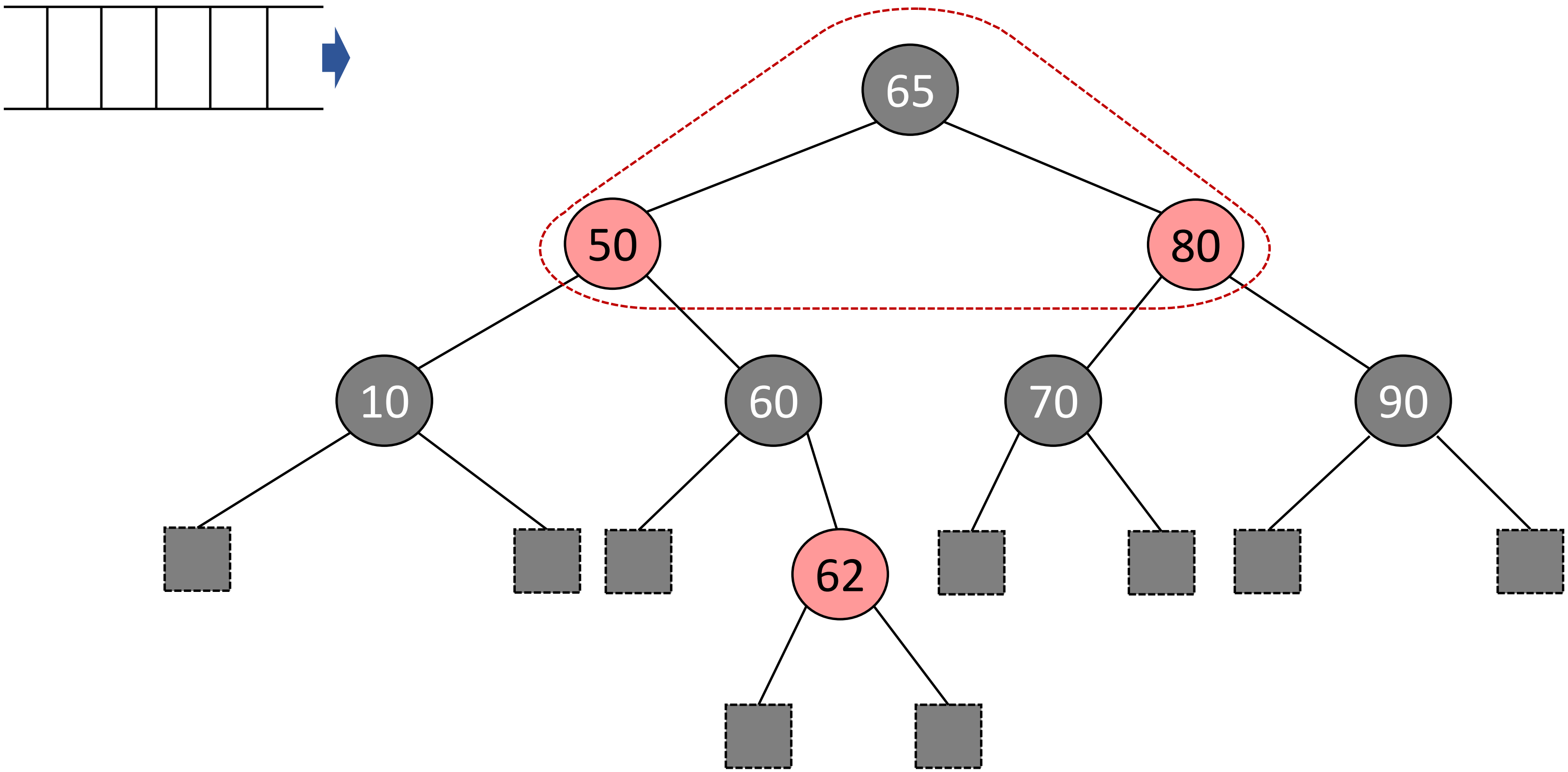


65 becomes the parent of 50 and 80

Example



Example

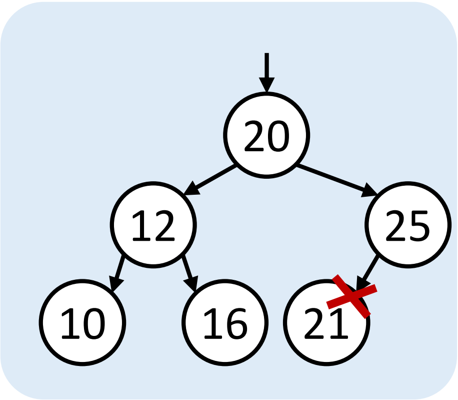


Red-Black Tree Operations

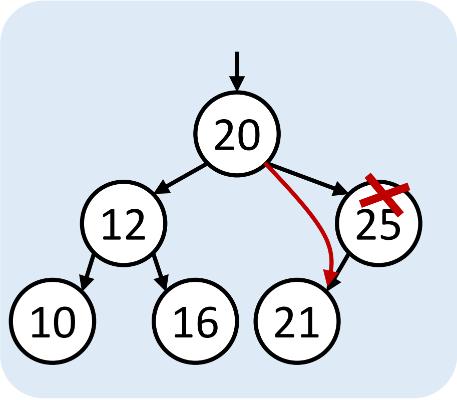
- Search
- Insertion
- **Deletion**

Standard BST Deletion

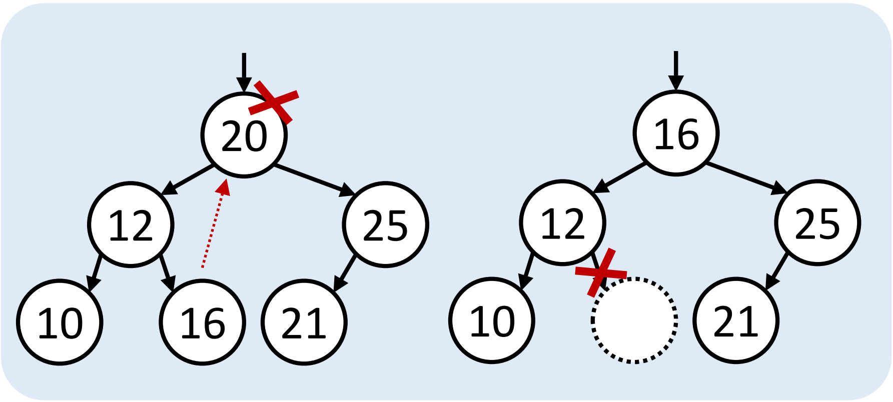
- Node to delete may have
  - no children→ just delete it
  - single child → bypass the node
  - two children →
    - Replace the node with its successor (or predecessor)
      - Successor is the largest node of the left subtree
    - Continue to delete the successor
- So we conclude that all the three cases end up with deleting a node with zero or one child



No children



One child

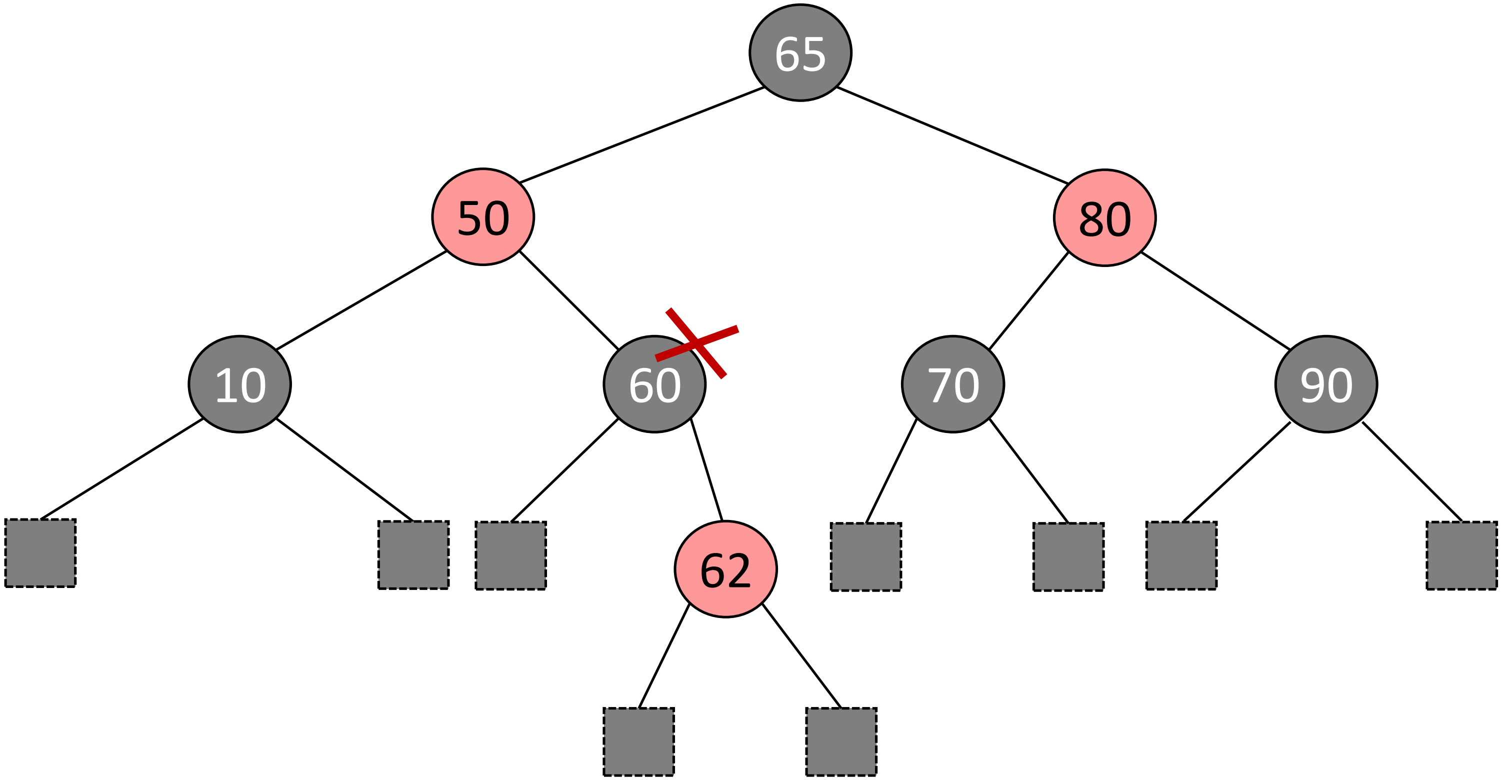


Two children

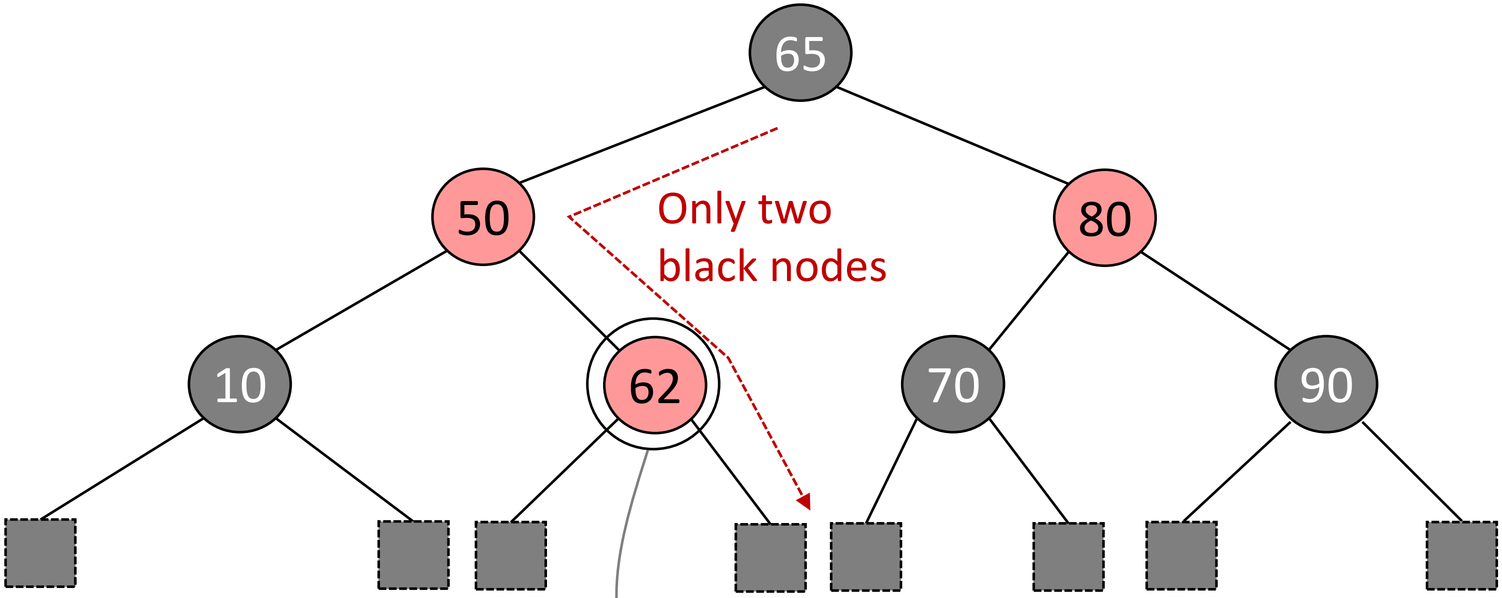
Red Black Tree Deletion

- Steps
  - Perform standard BST deletion, which ends up with deleting a node (the victim) with zero or one child
  - If the victim is red, deletion is done
    - Deleting a red victim cannot violate red-black properties
  - If the victim is black, tree rebalancing is required
    - Deleting a black victim always violates red-black properties
      - Number of black nodes among all root-to-nil paths must become unequal

Example 1

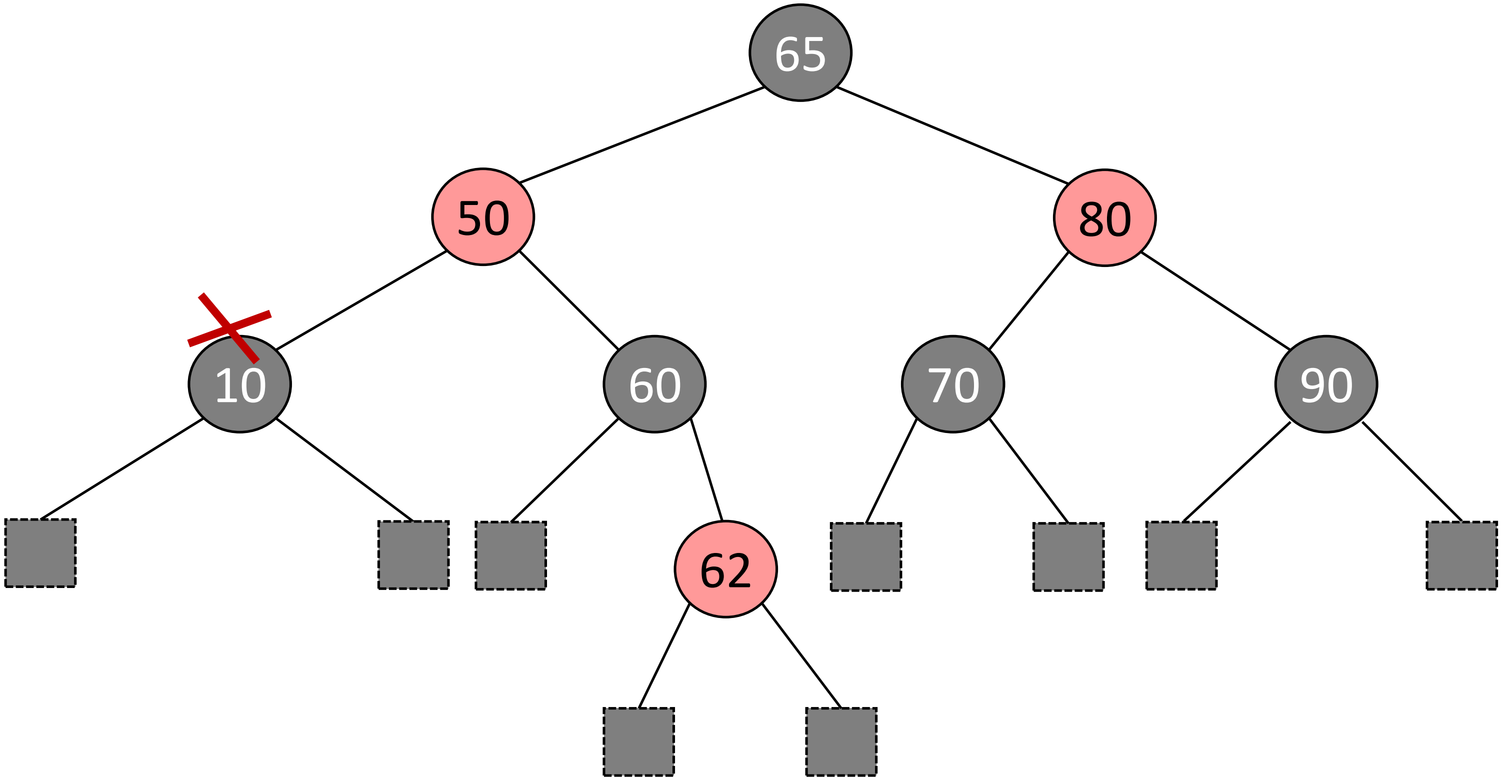


Example 1

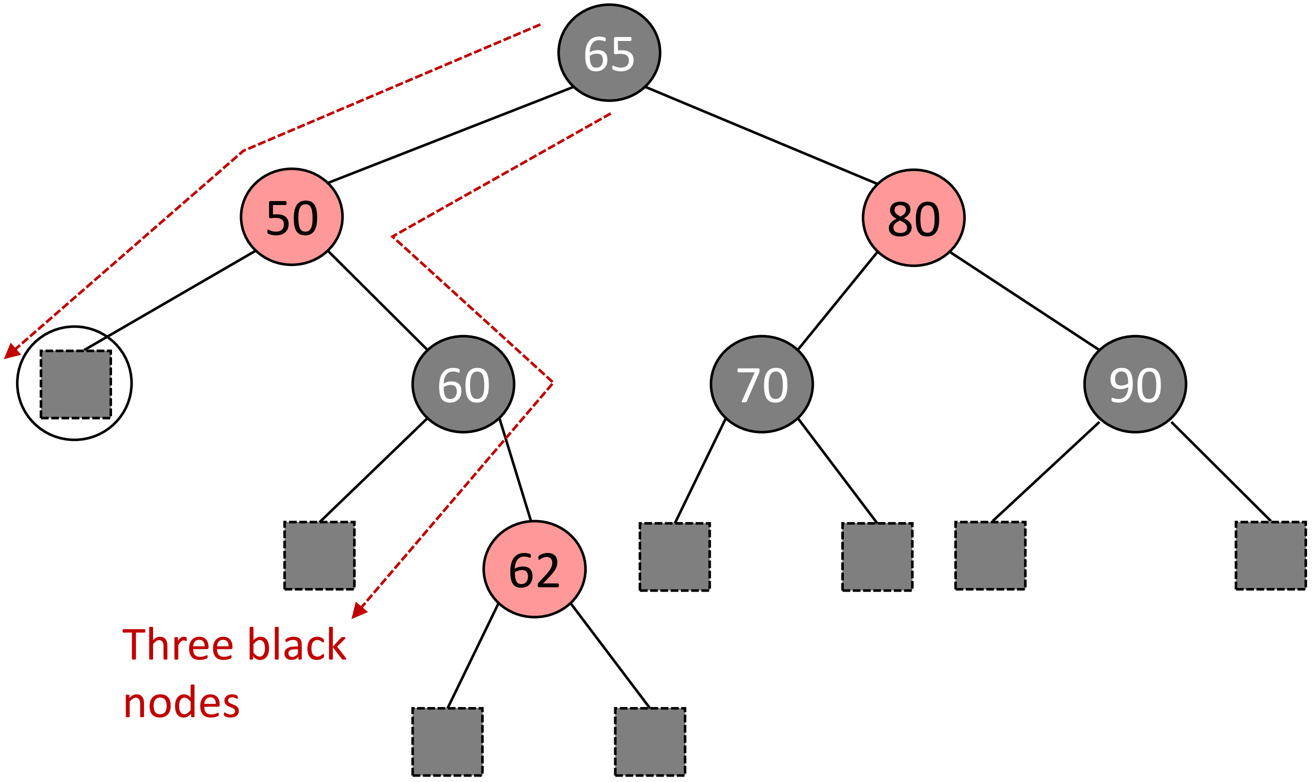


A black circle is added to the node that moves to the position of the deleted black node
代表該位置缺一個黑色的node

Example 2



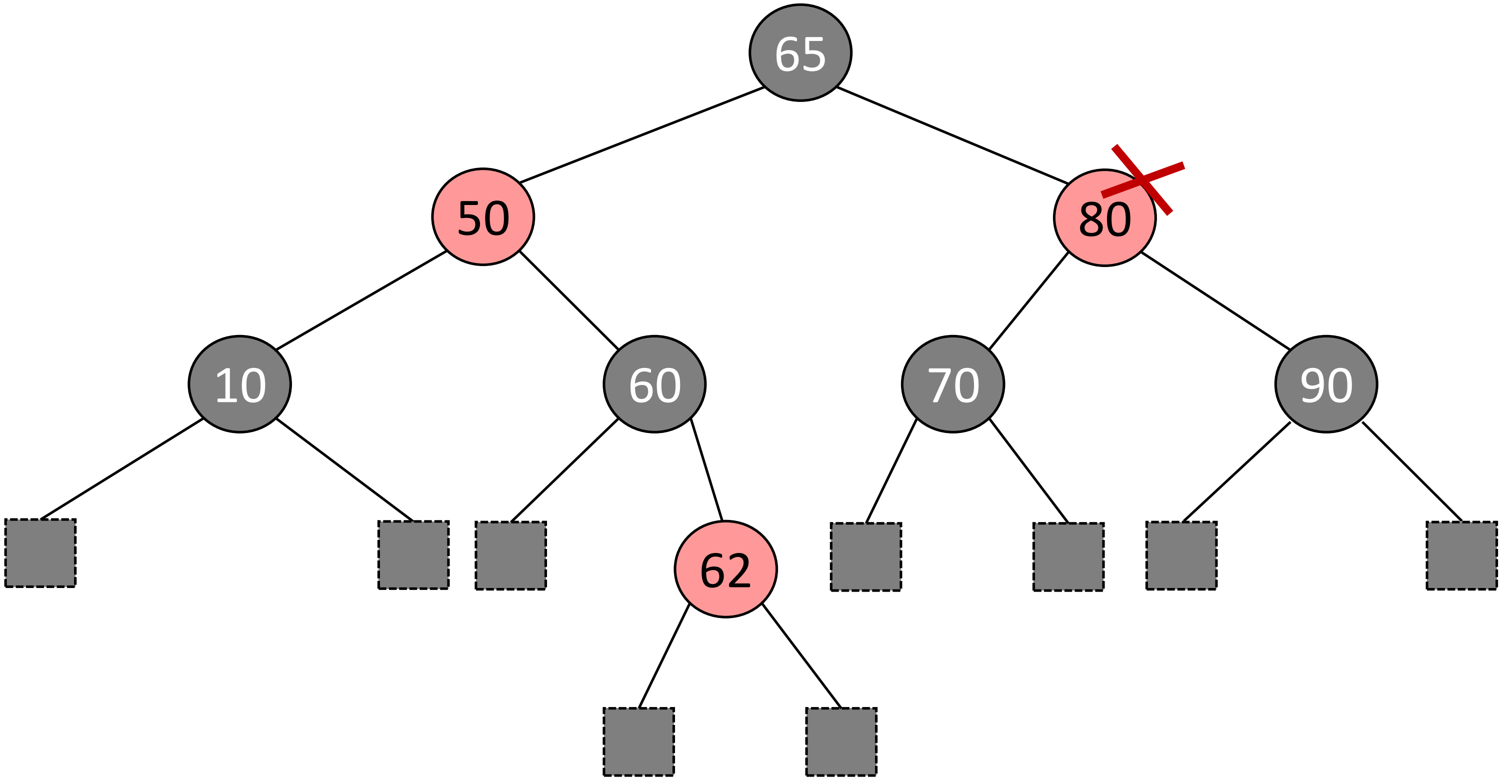
Example 2



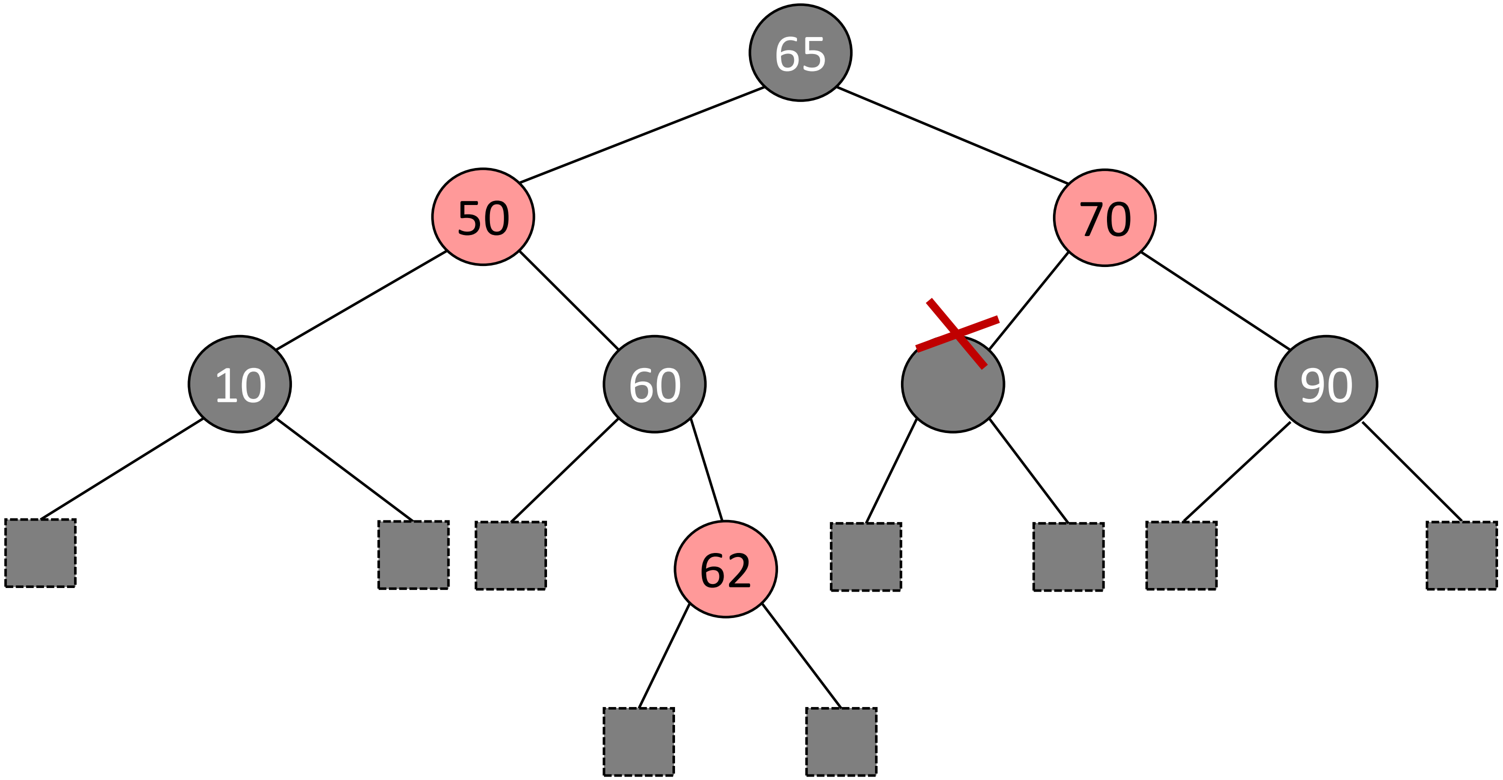
Only two black nodes

Three black nodes

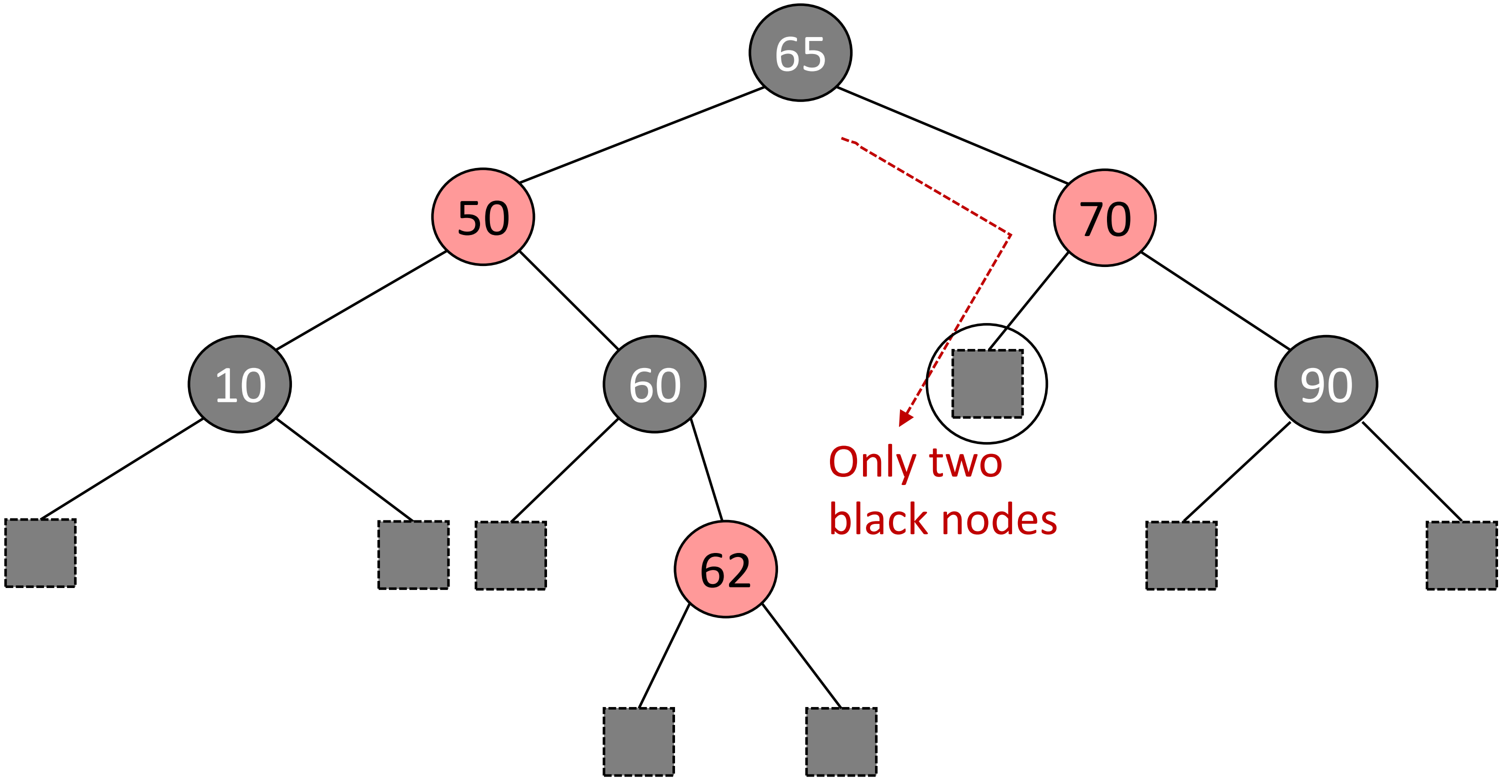
Example 3



Example 3

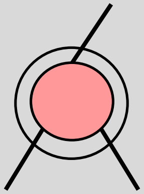


Example 3

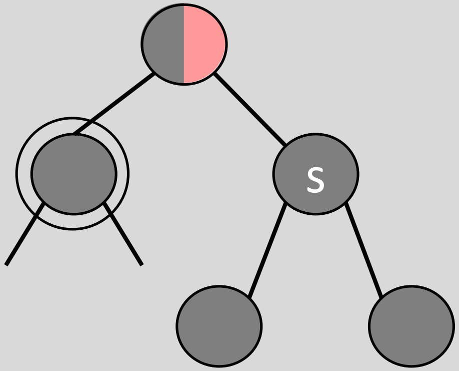


Only two black nodes

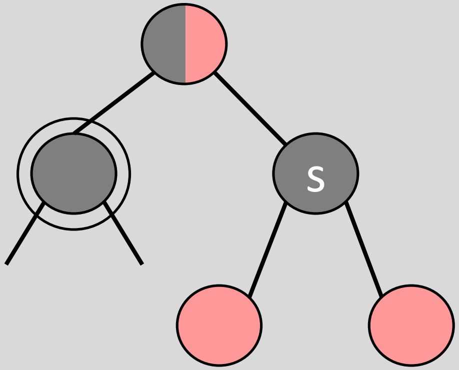
Four Cases



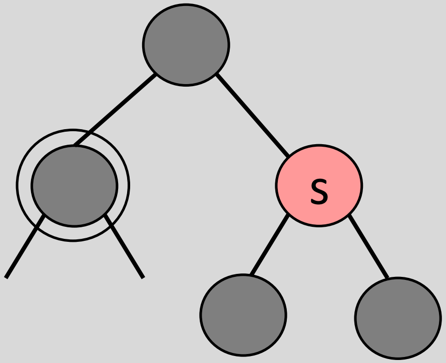
The circled node is red



The circled node is black, its sibling is black, and its sibling has two black children



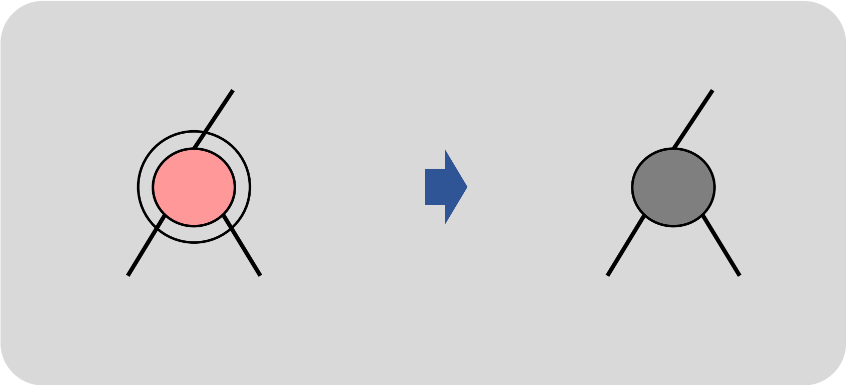
The circled node is black, its sibling is black, and its sibling has at least one red child.



The circled node is black, and its sibling is red

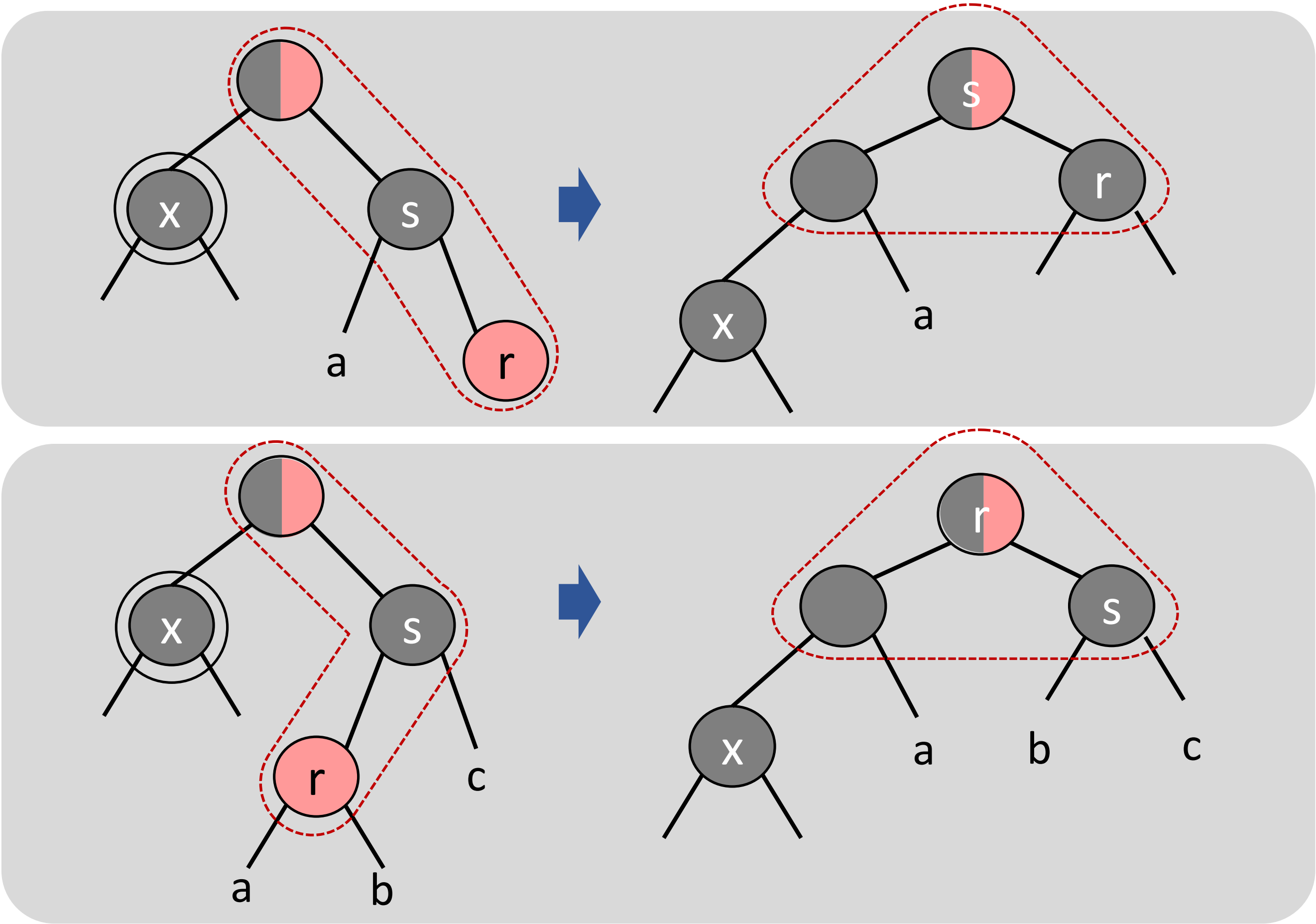
Case 1

- The circled node is red
- Changing the node color from red to black



Case 2

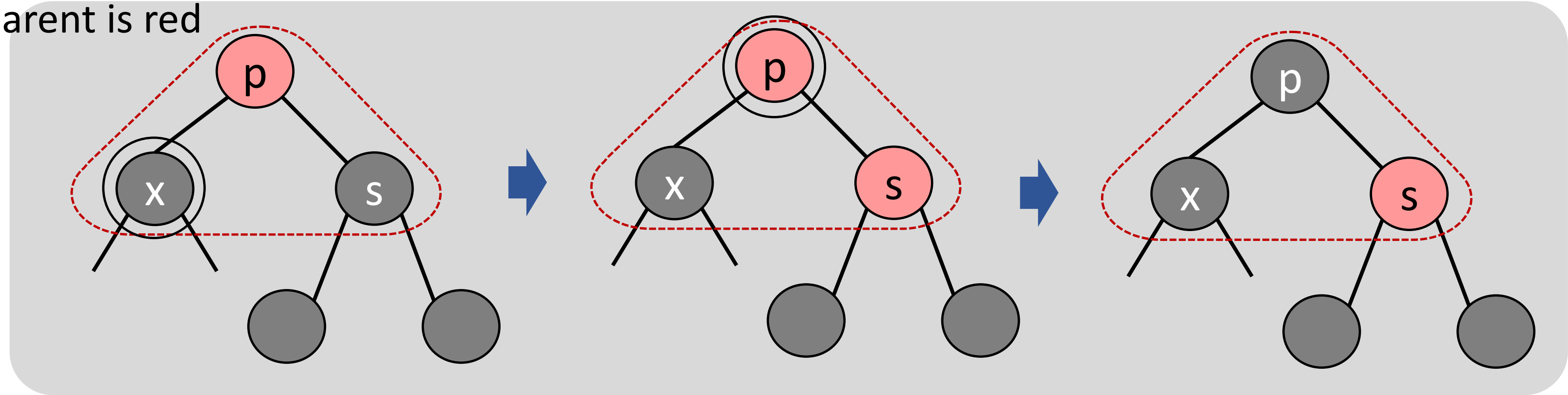
- The circled node is black, its sibling is black, and its sibling has at least one red child
- Perform rotation



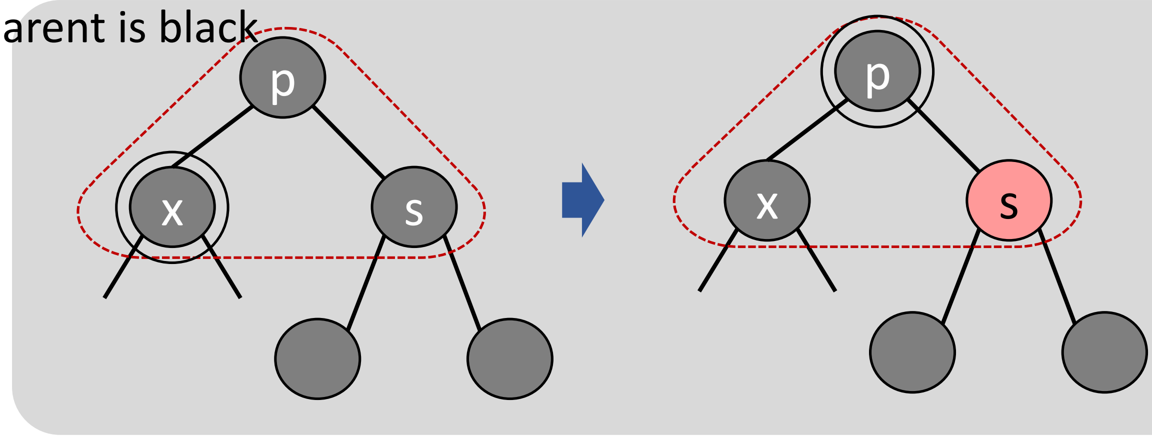
Case 3

- The circled node is black, its sibling is black, and its sibling has two black children
- Perform color changes to move the black circle one level up the tree

If the parent is red



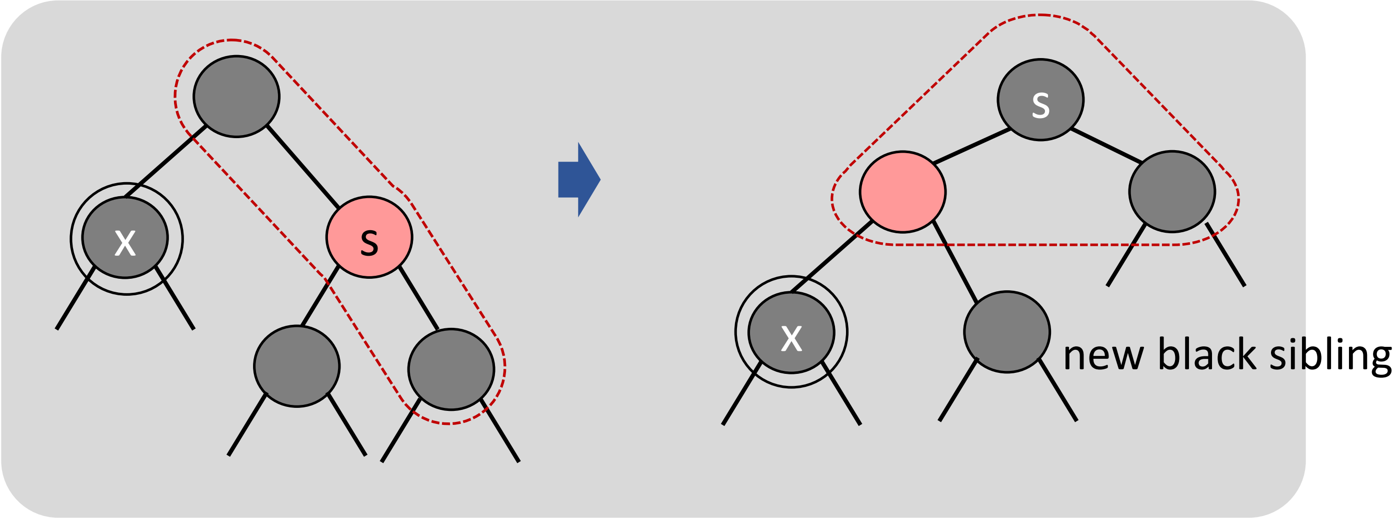
If the parent is black



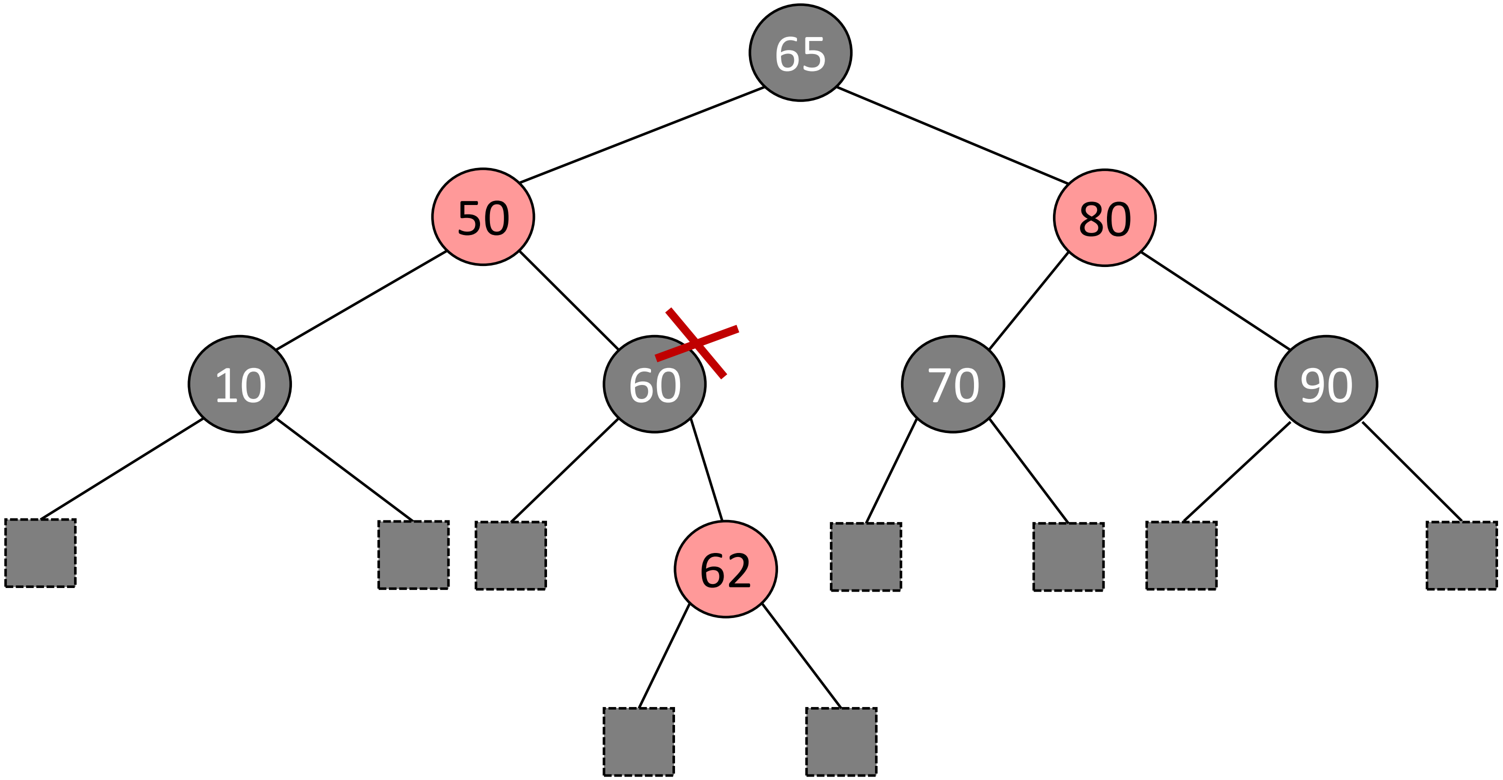
(need further rebalancing)

Case 4

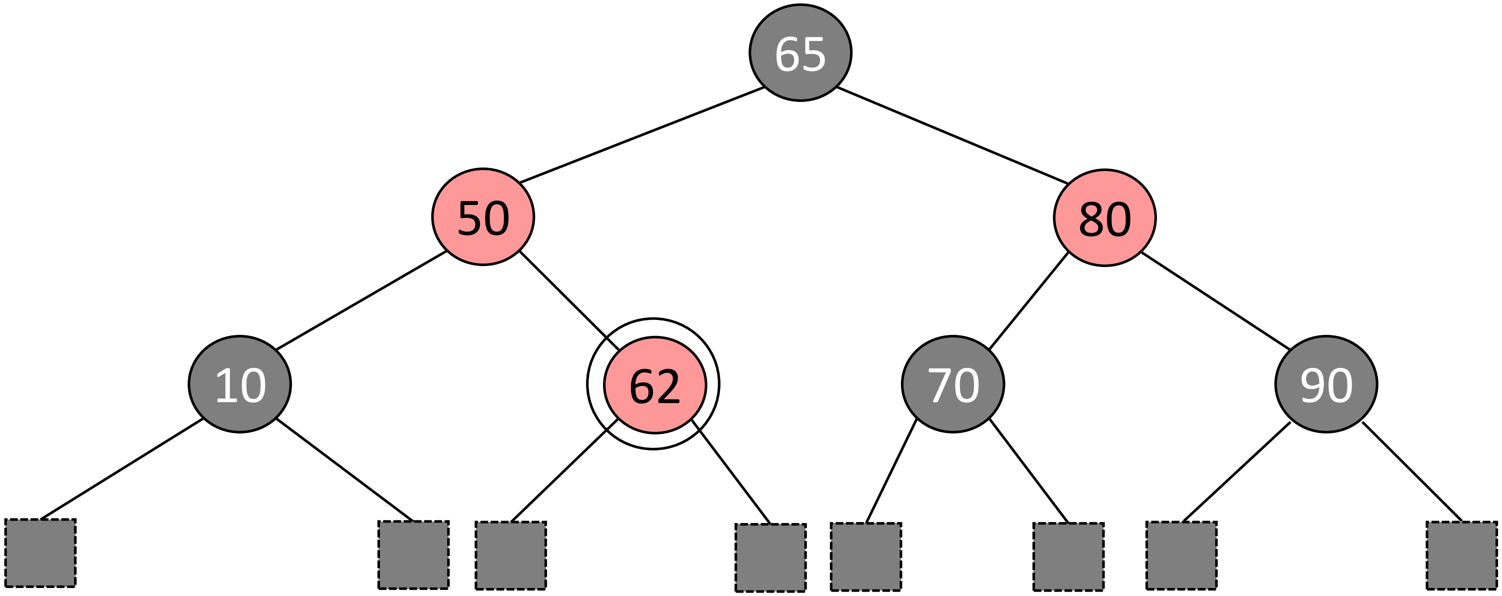
- The circled node is black, and its sibling is red
- Perform rotation to make a black node to become a new sibling of the circled node
  - Then the case becomes case 2 or case 3



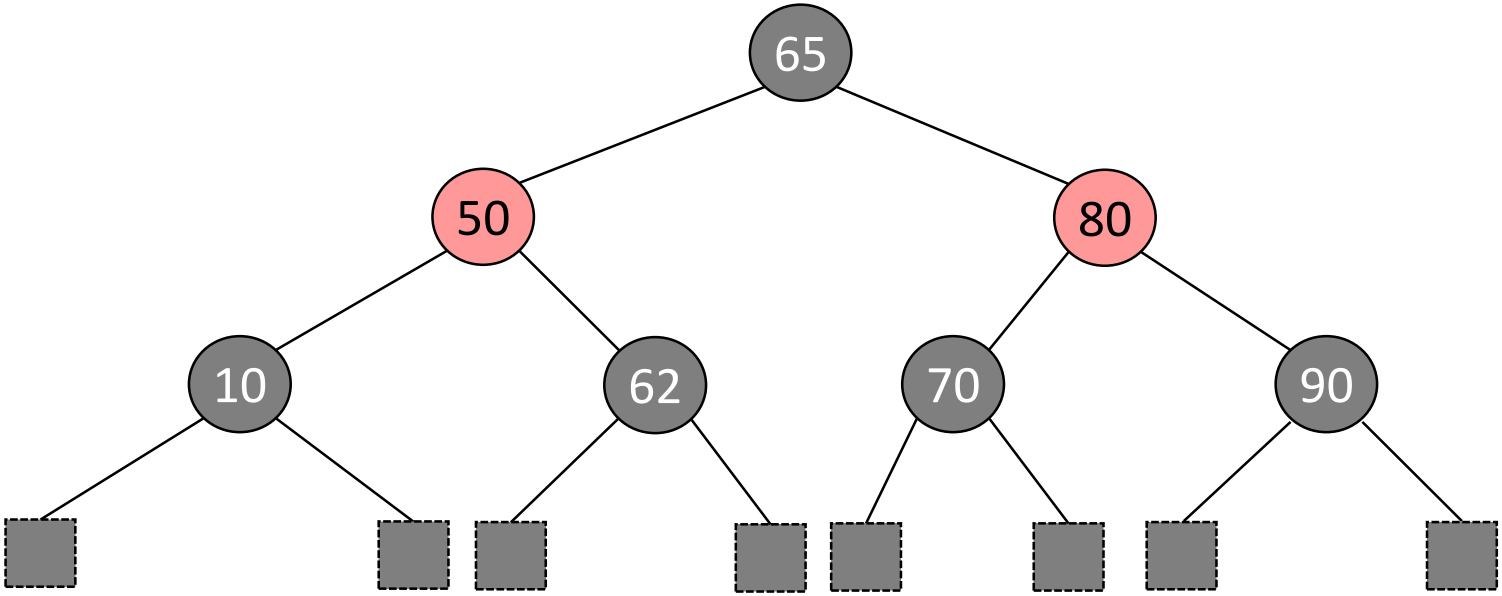
Example 1



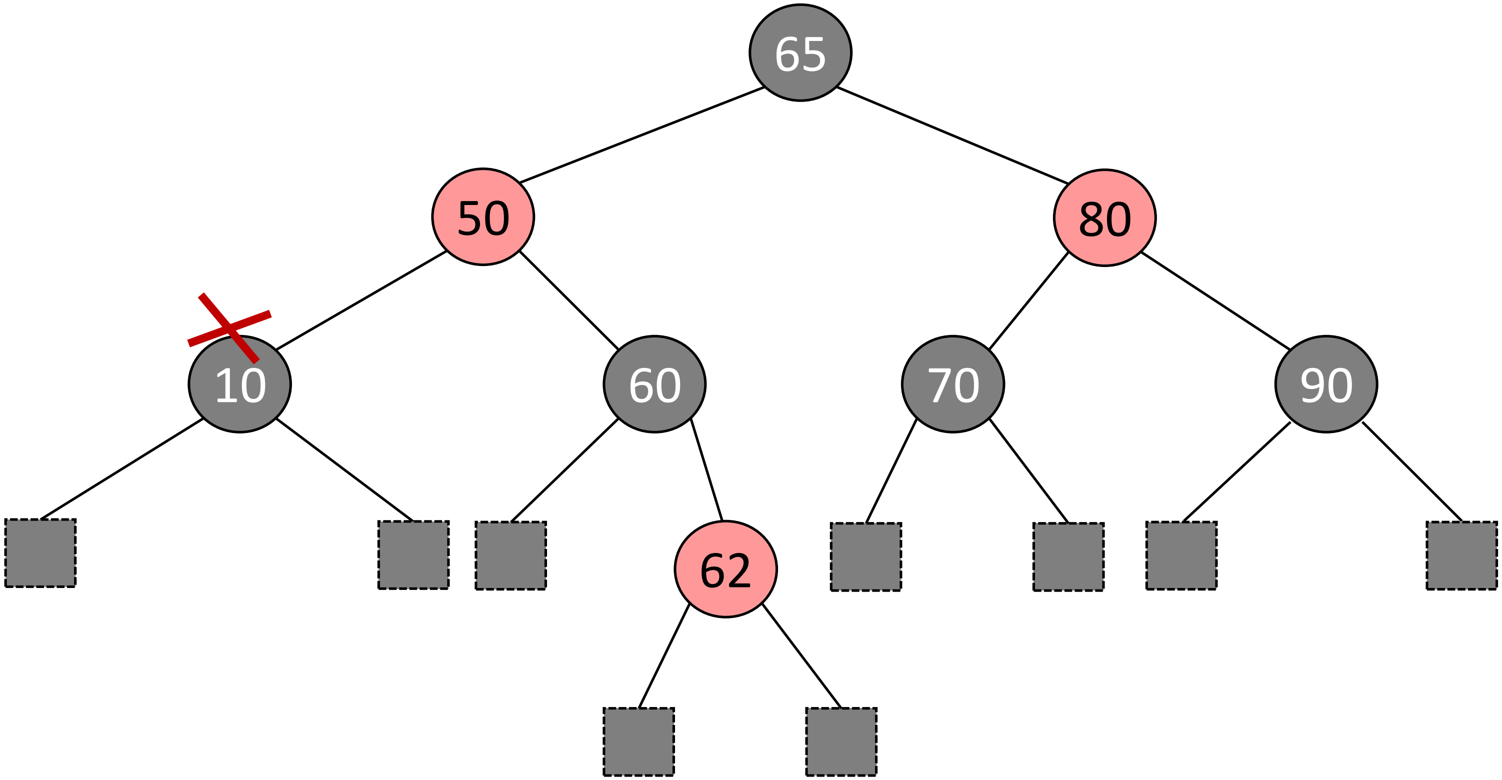
Example 1



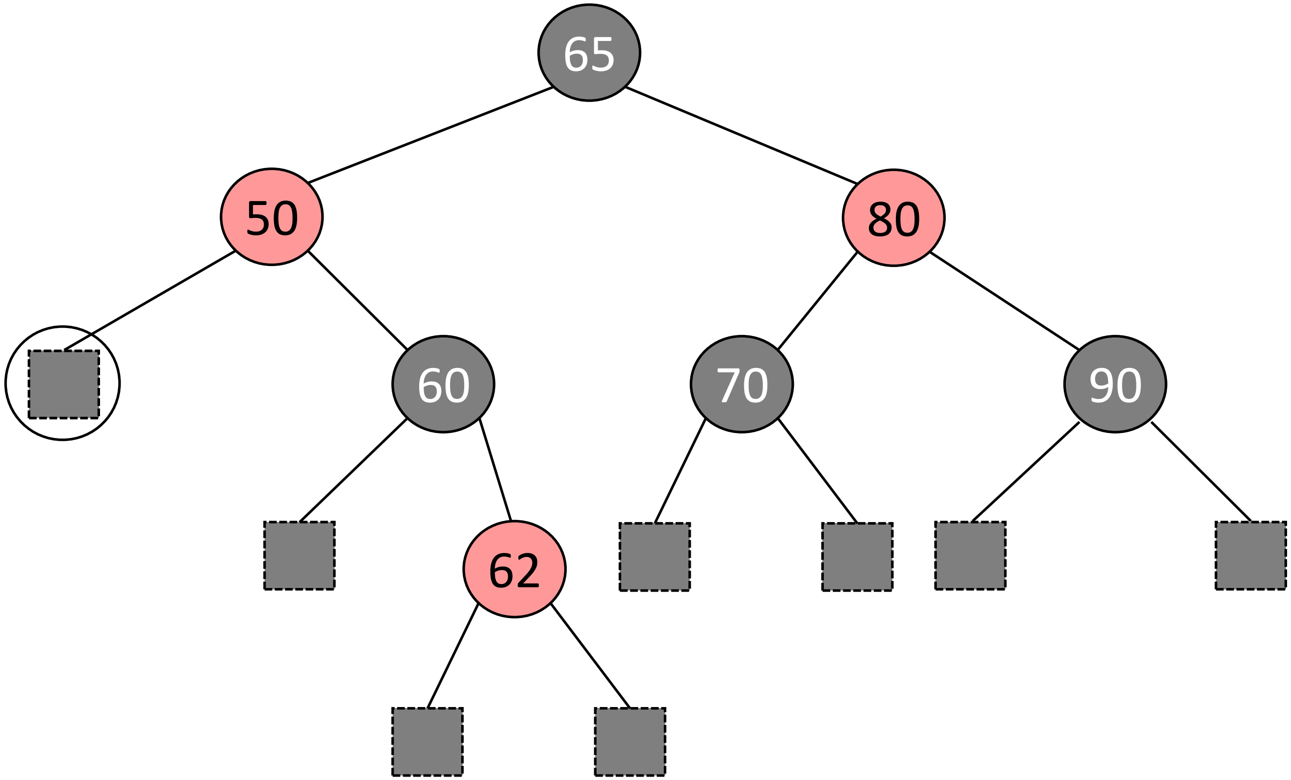
Example 1



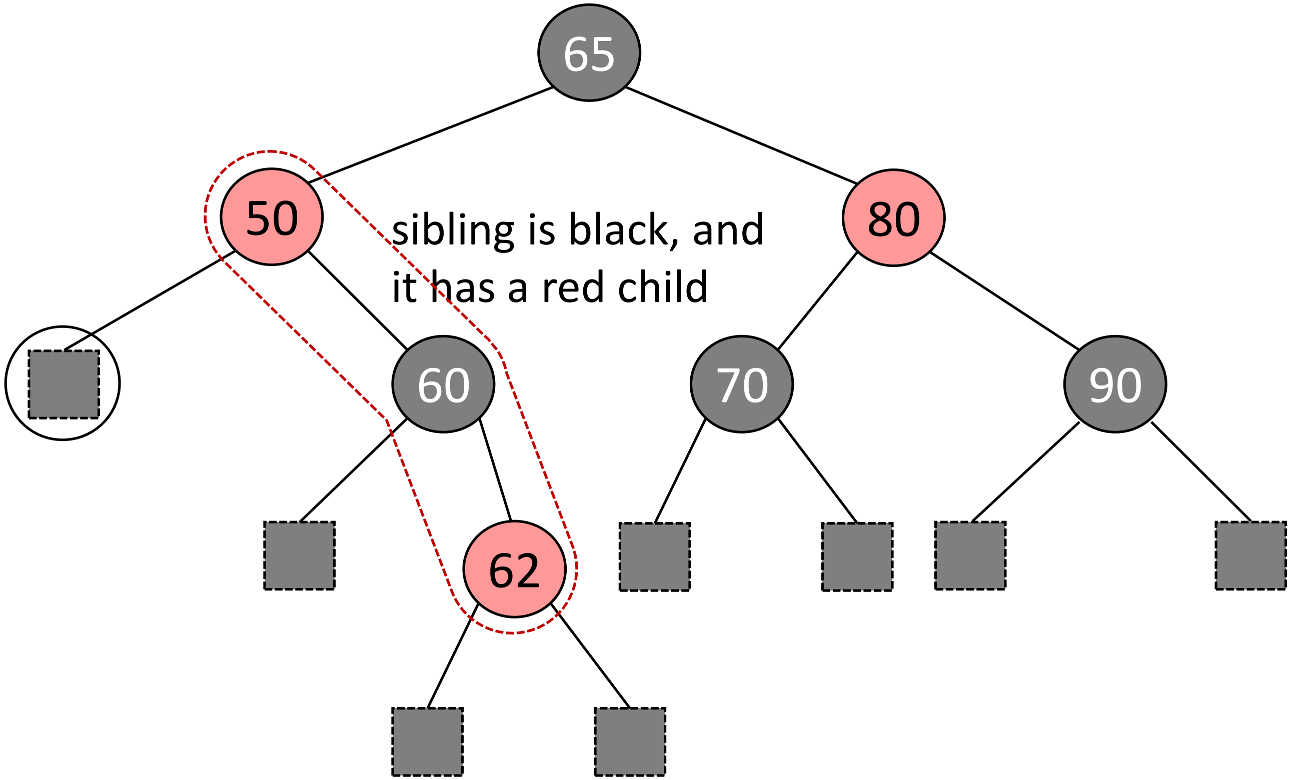
Example 2



Example 2

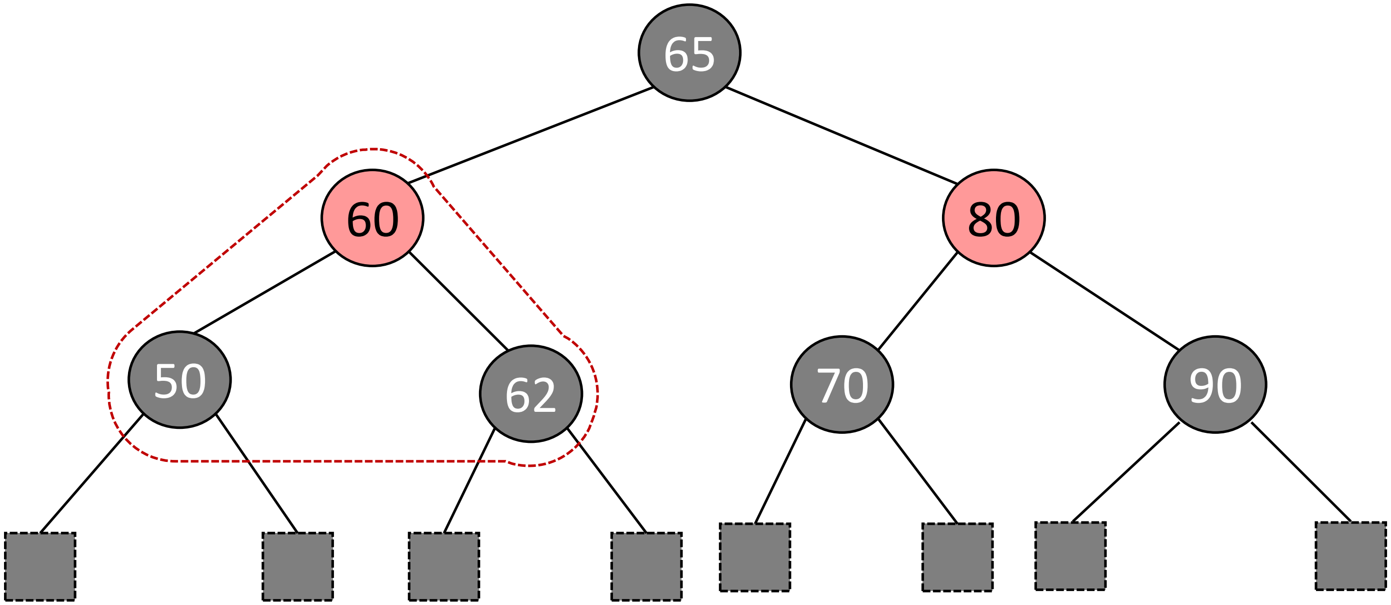


Example 2

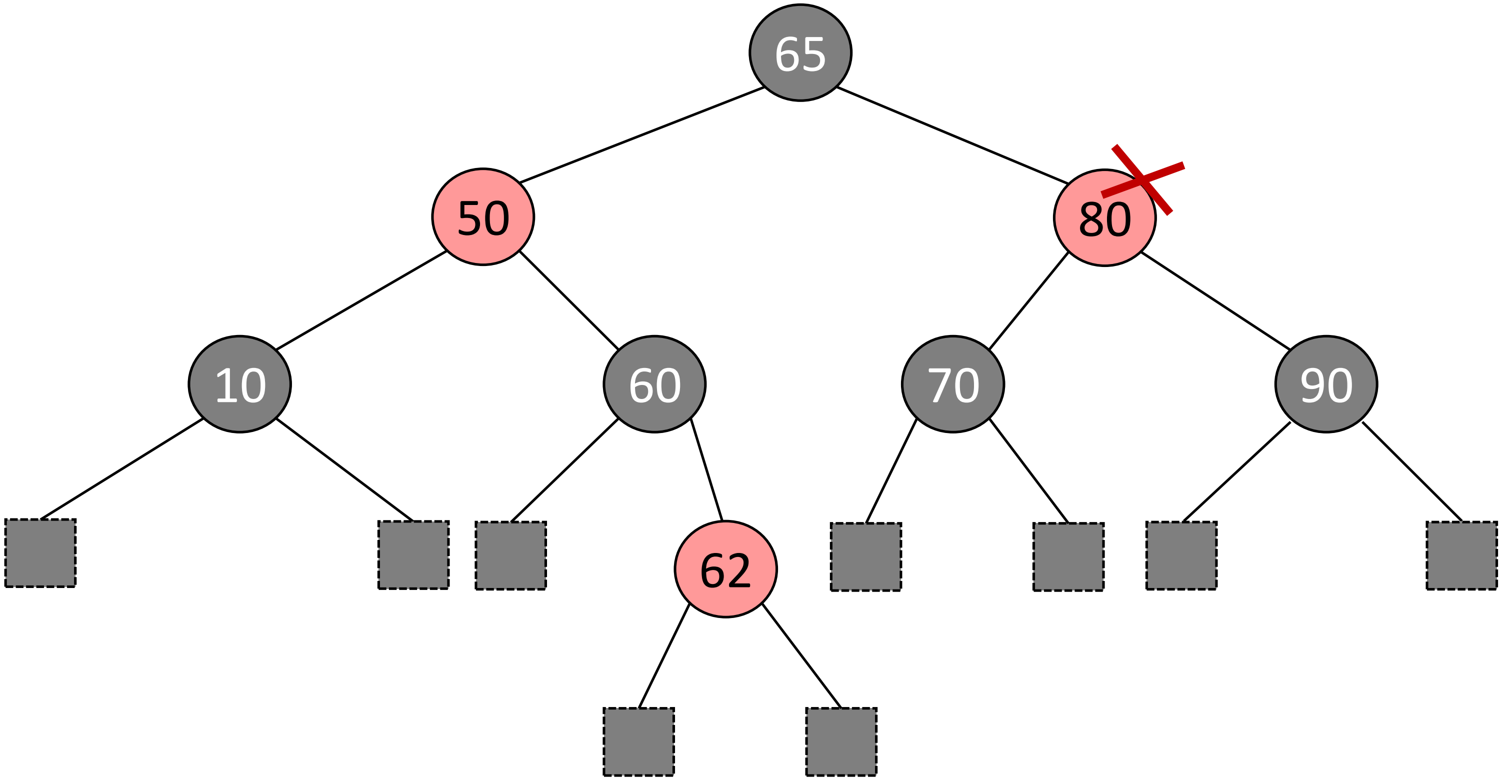


sibling is black, and it has a red child

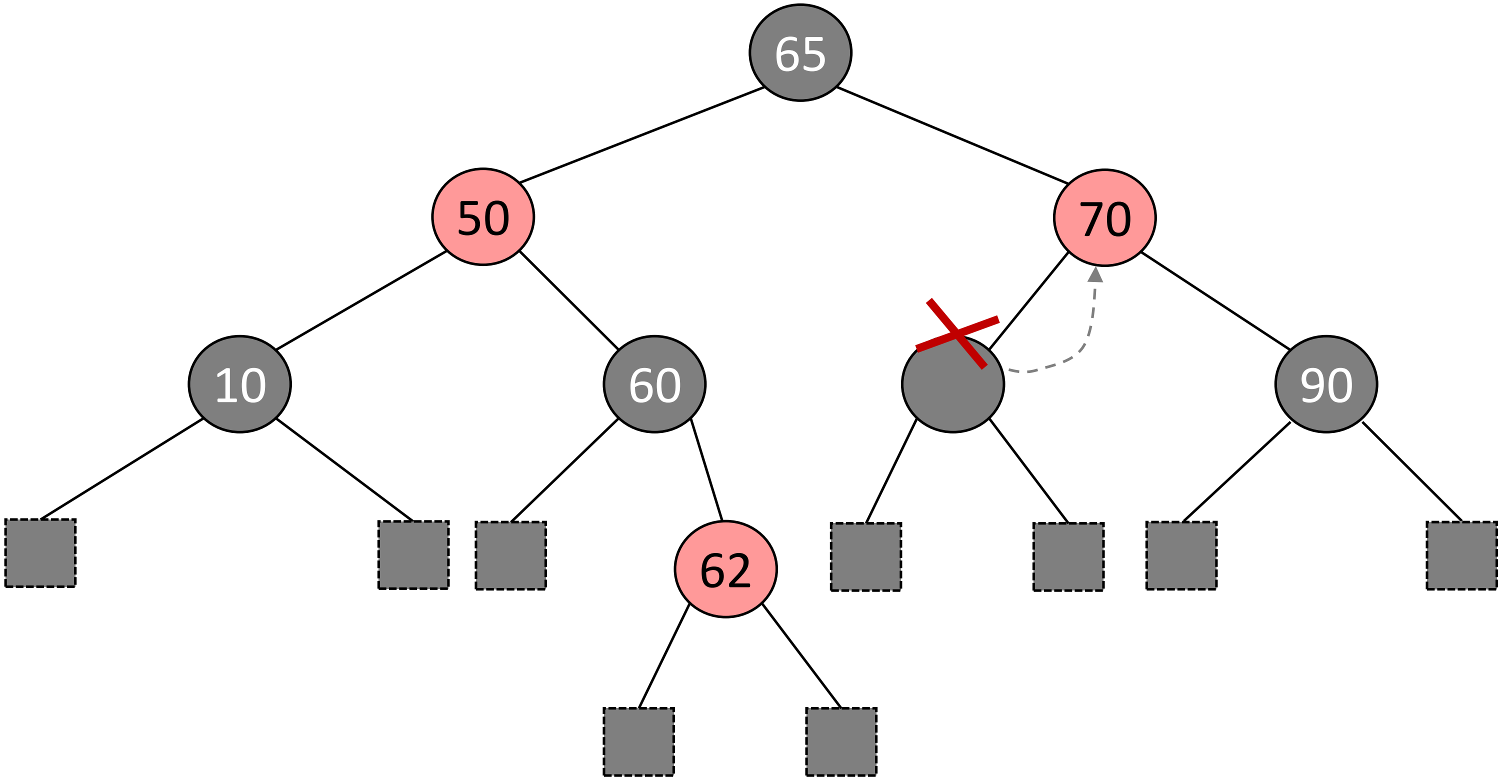
Example 2



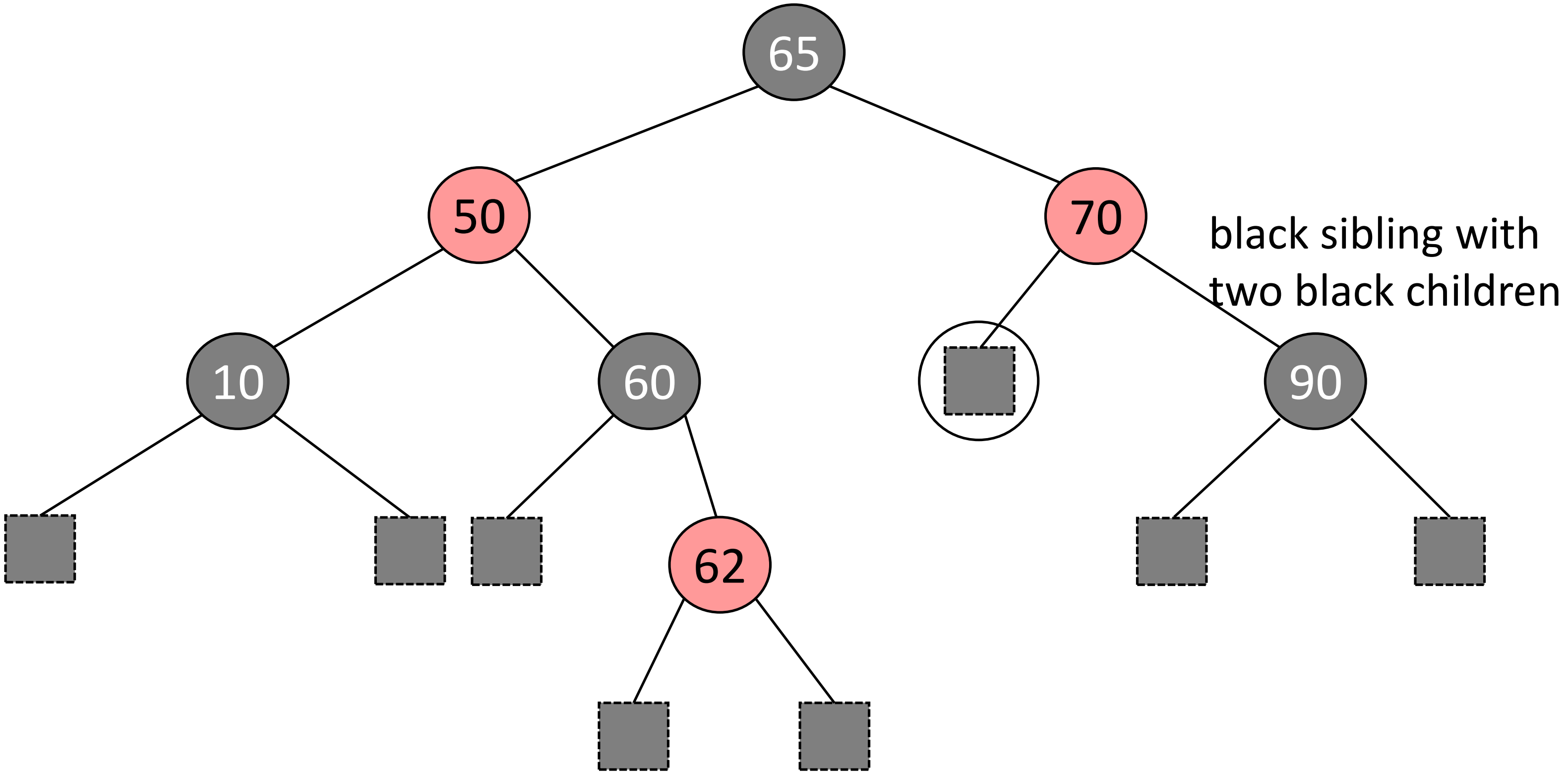
Example 3



Example 3

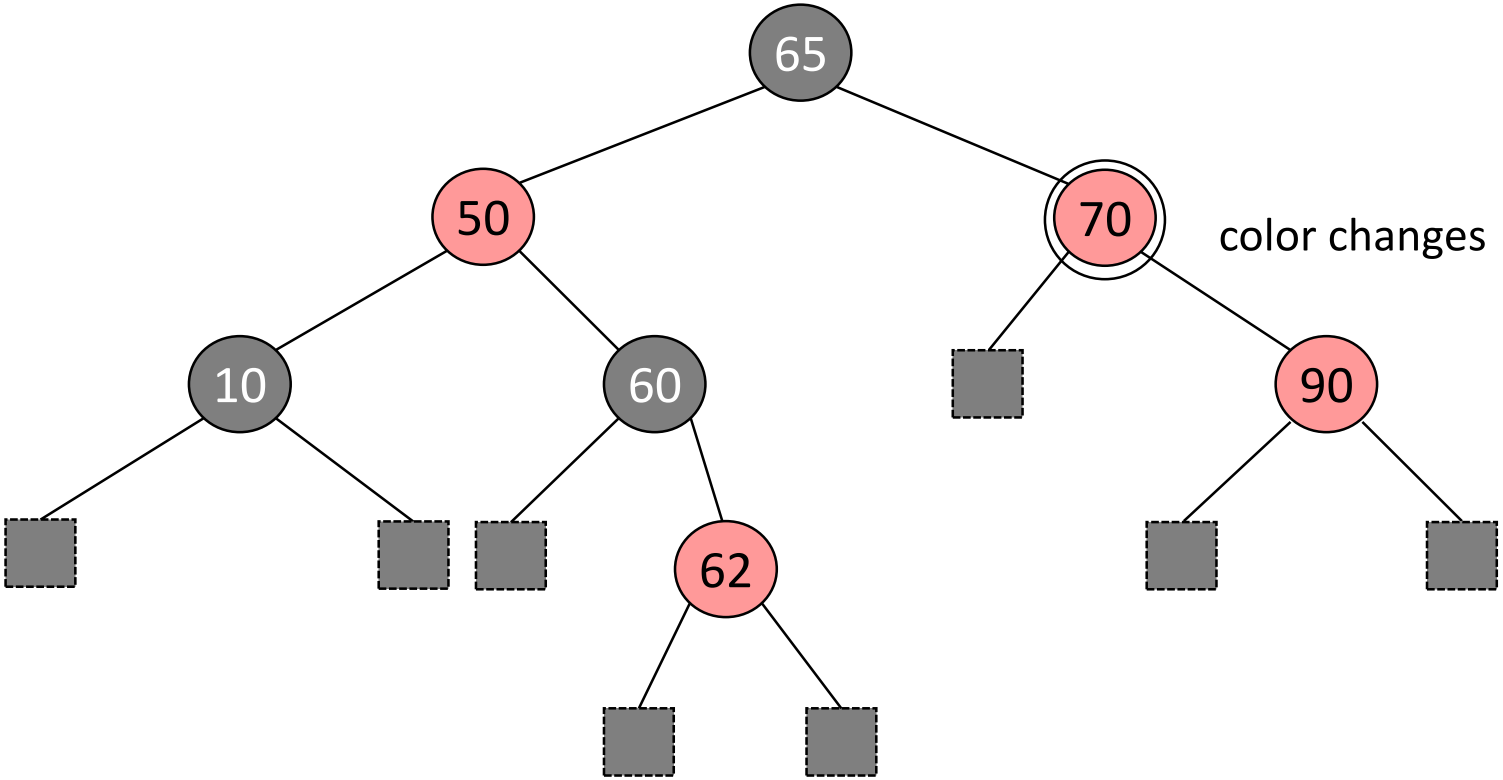


Example 3



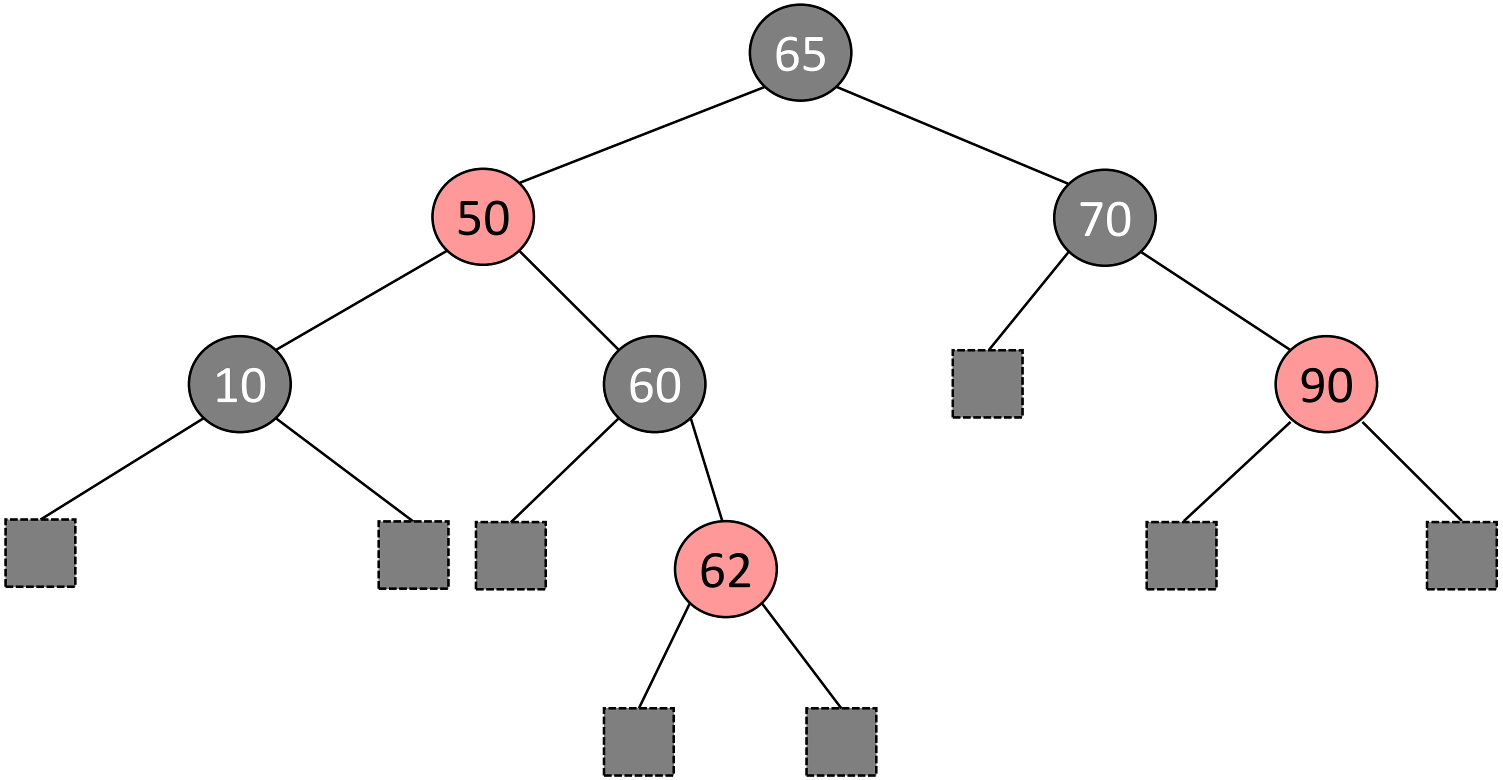
black sibling with two black children

Example 3

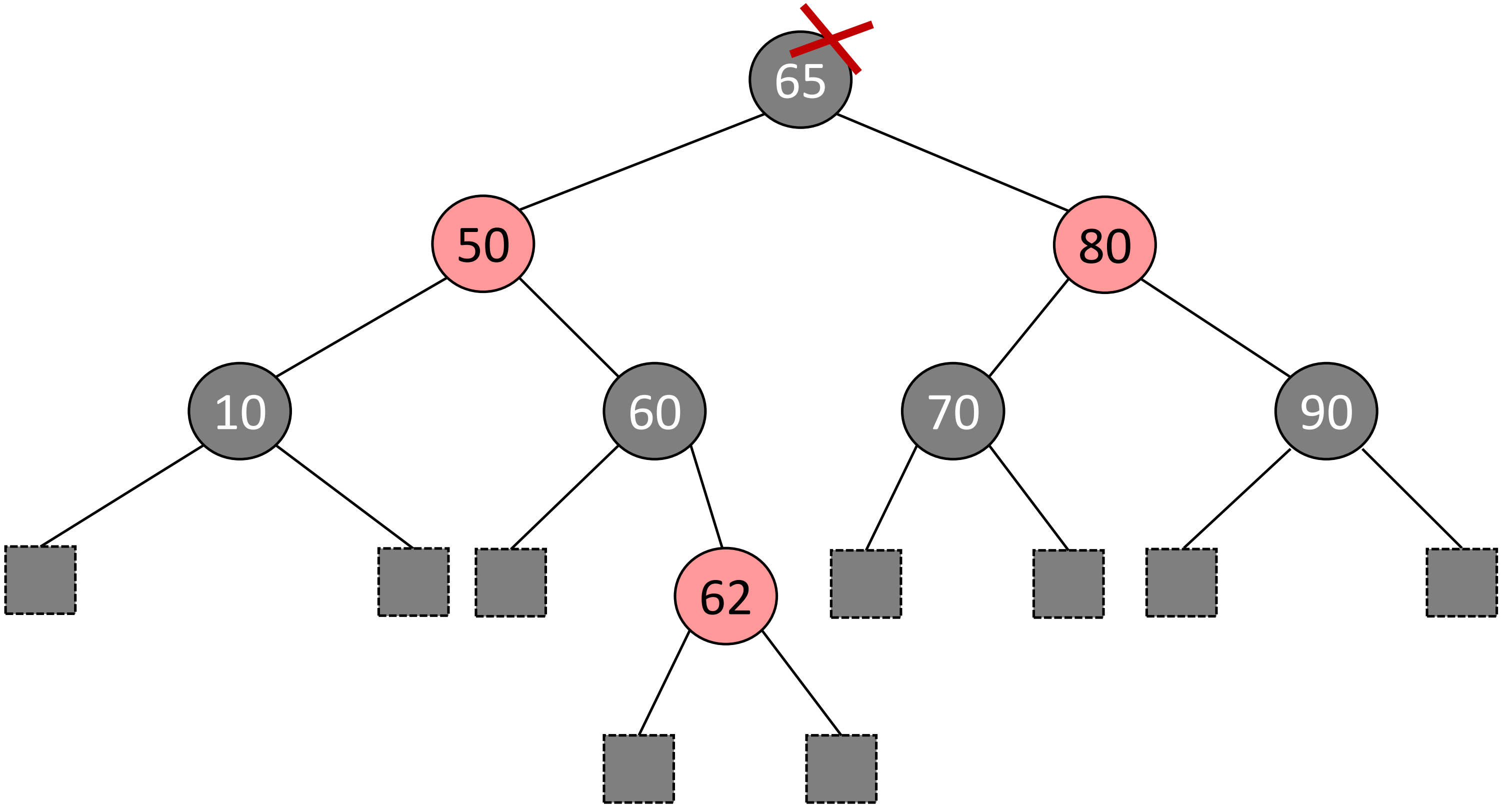


color changes

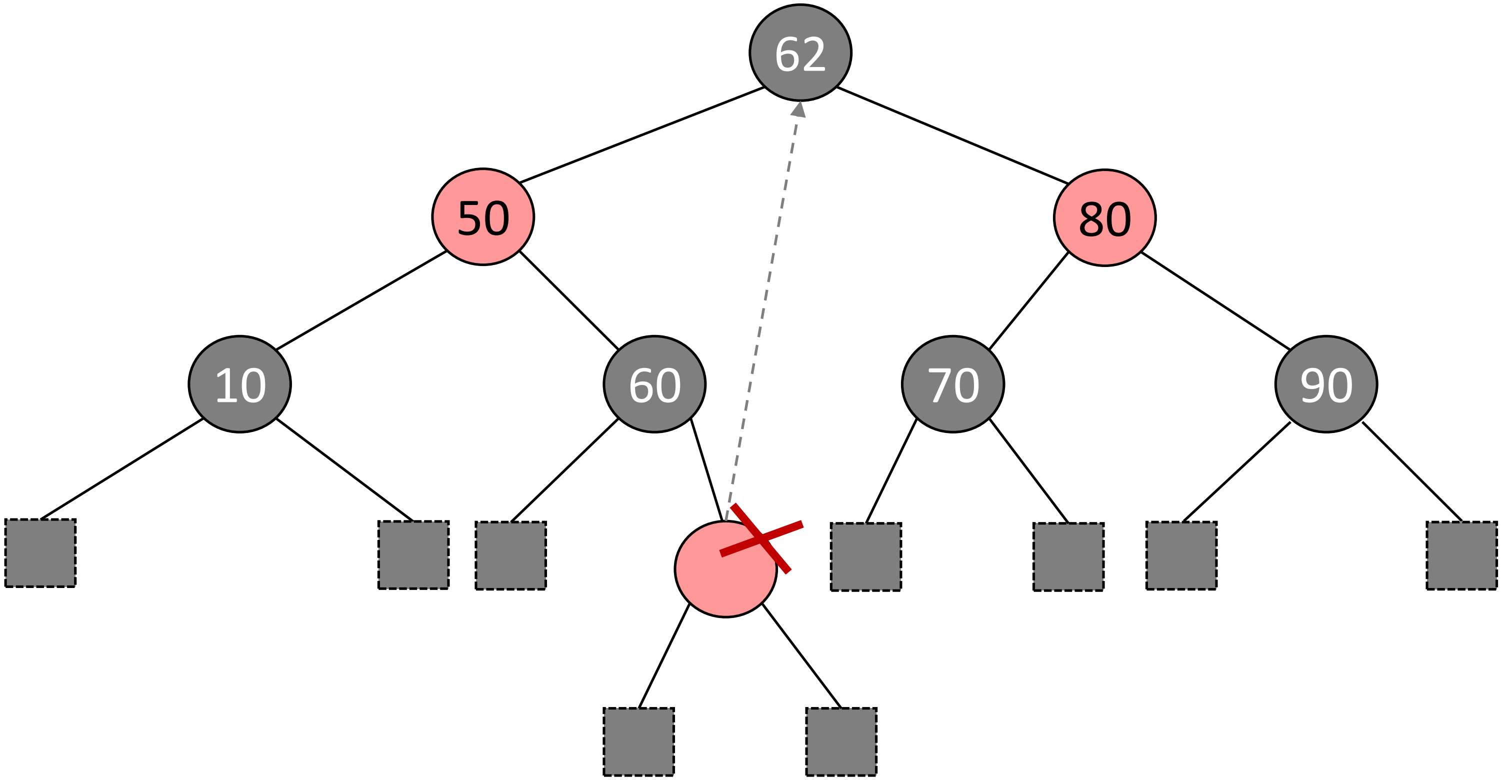
Example 3



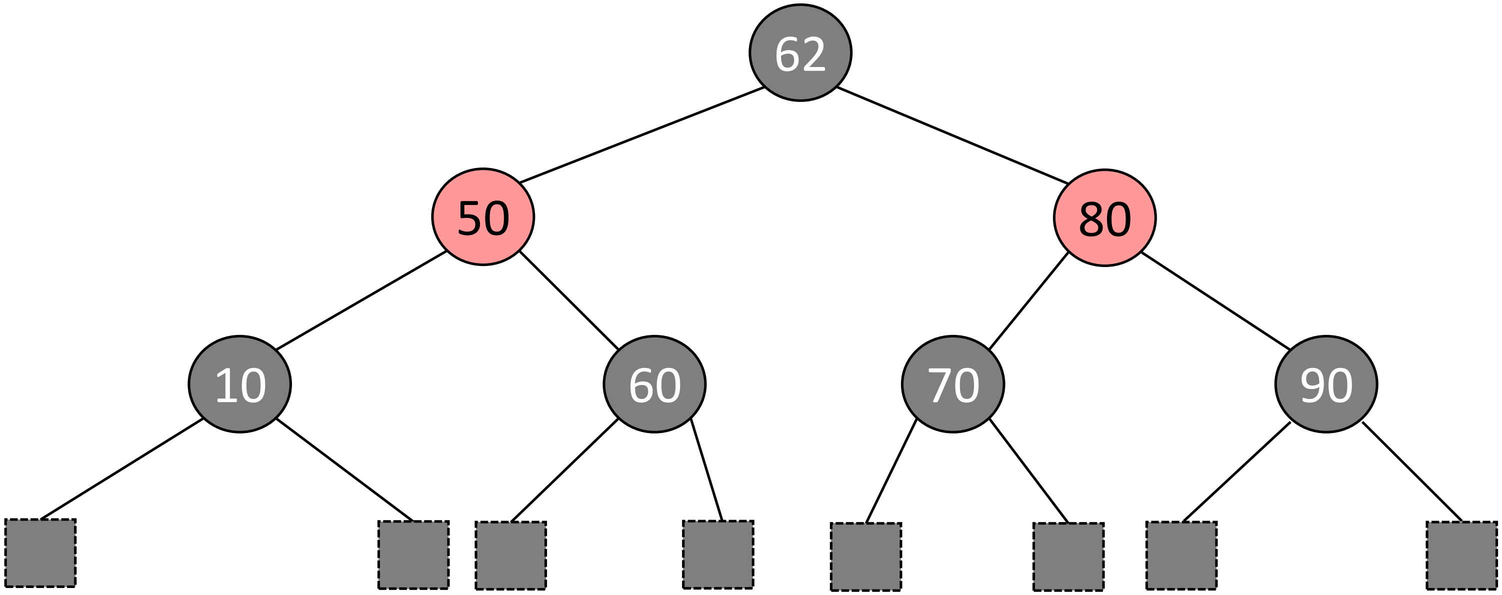
Example 4



Example 4



Example 4



祝 暑假快樂
身體健康
學業順利