CS302 - Data Structures

using C++

Topic: Balanced Search Trees

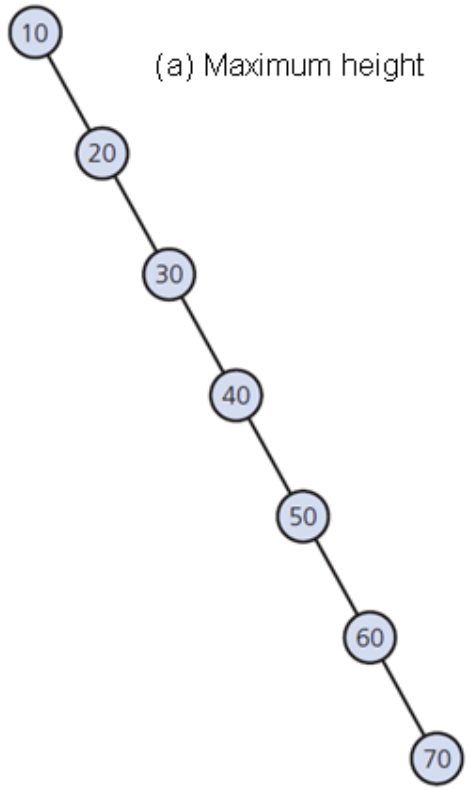
Kostas Alexis

Balanced Search Trees

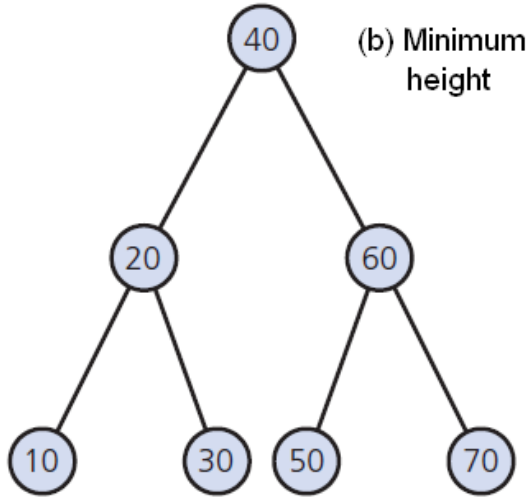
- Height of binary search tree
  - Sensitive to order of additions and removals
- Various search trees can retain balance
  - Despite additions and removals

Balanced Search Trees

The tallest and shortest binary search trees containing the same data



(a) Maximum height



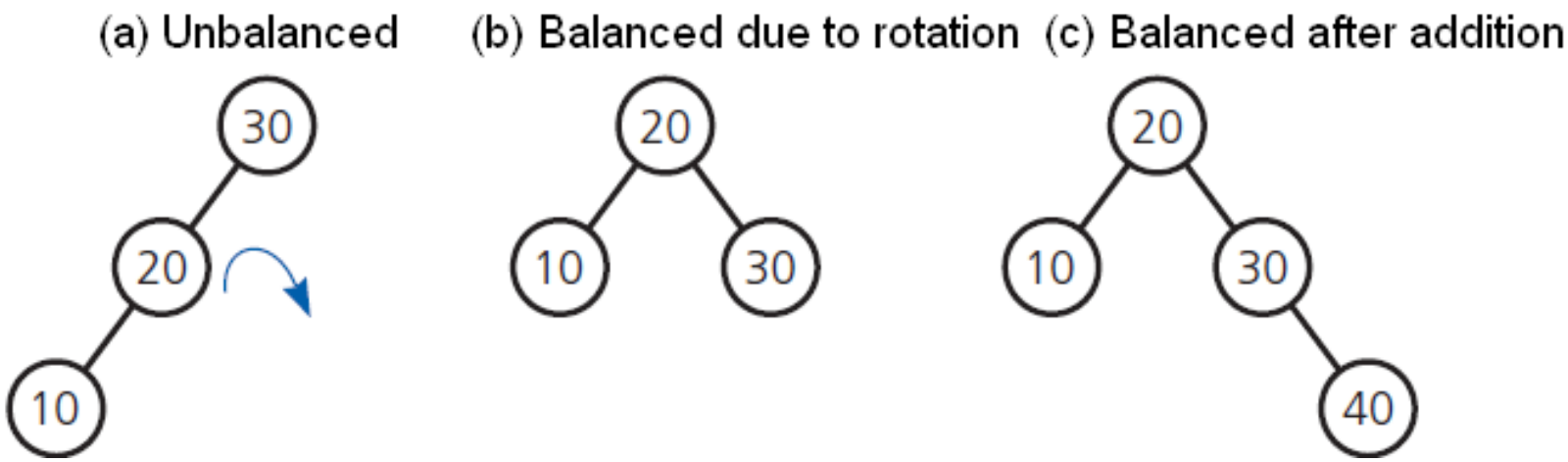
(b) Minimum height

AVL Trees

- An AVL (**Adelson-Velskii and Landis**) tree
  - A balanced binary search tree
- Maintains its height close to the minimum
- Rotations restore the balance

AVL Trees

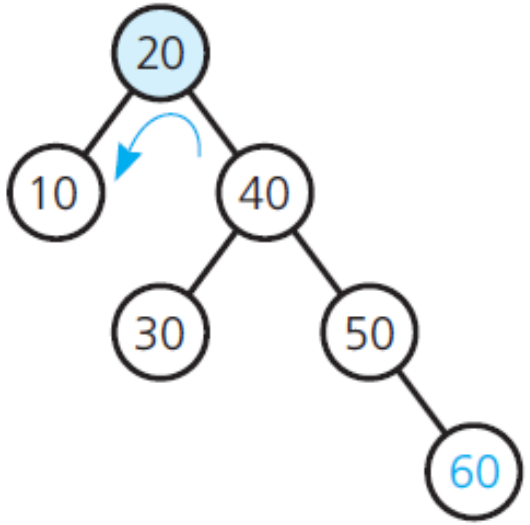
An unbalanced binary search tree



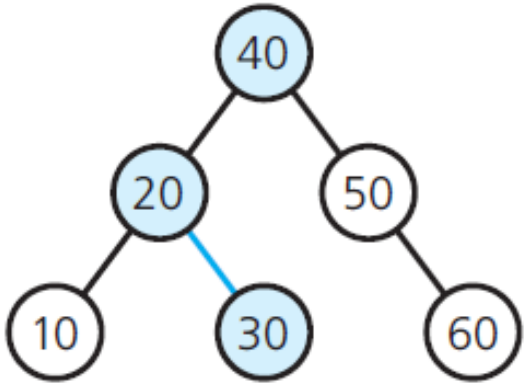
AVL Trees

Correcting an imbalance in an AVL tree due to an addition by using a single rotation to the left

(a) The addition of 60 to an AVL tree destroys its balance



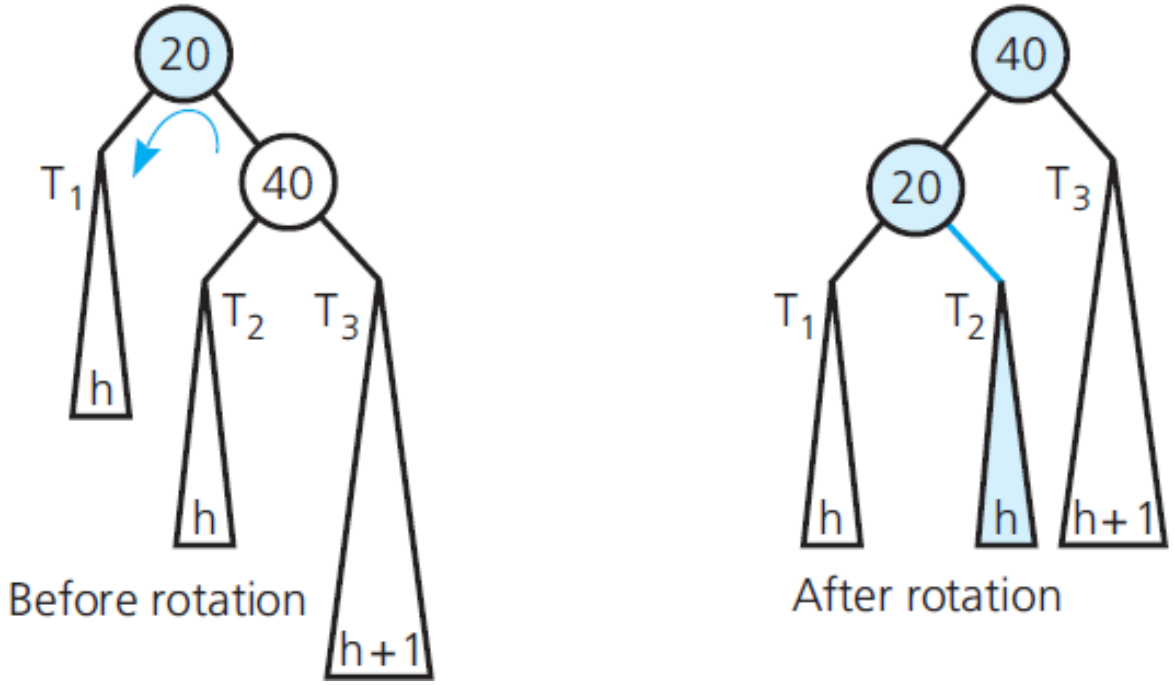
(b) A single left rotation restores the tree's balance



AVL Trees

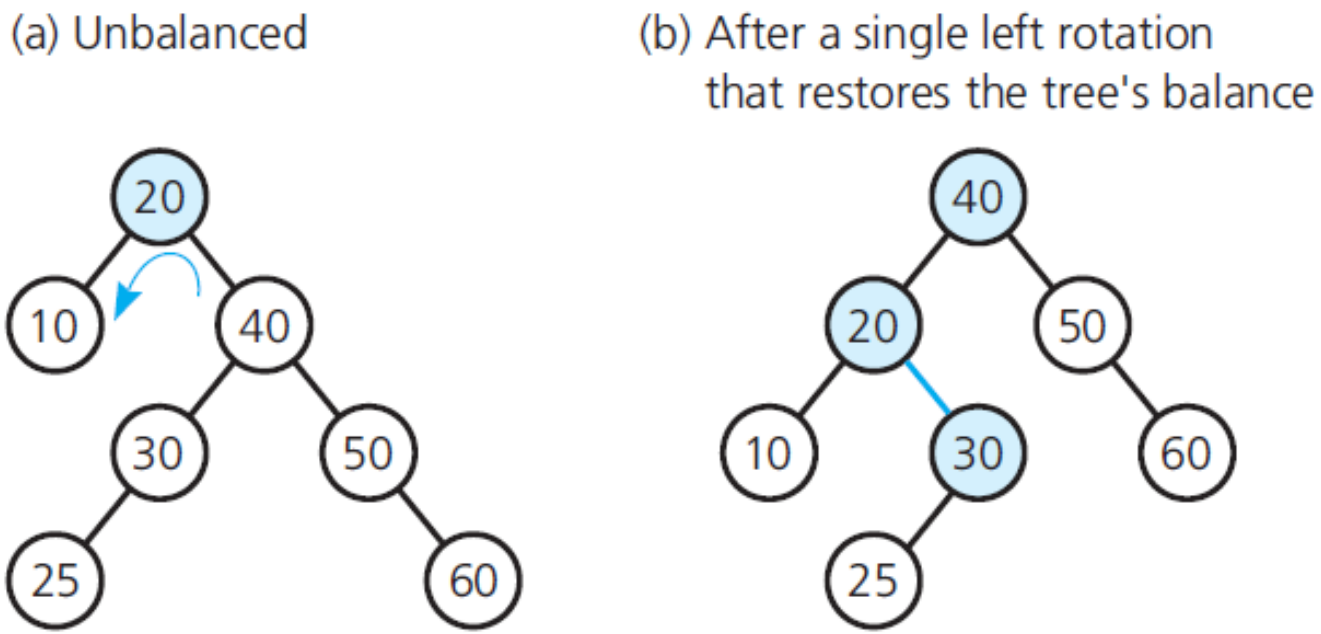
[Continued]

(c) The general case for a single left rotation in an AVL tree whose height decreases



AVL Trees

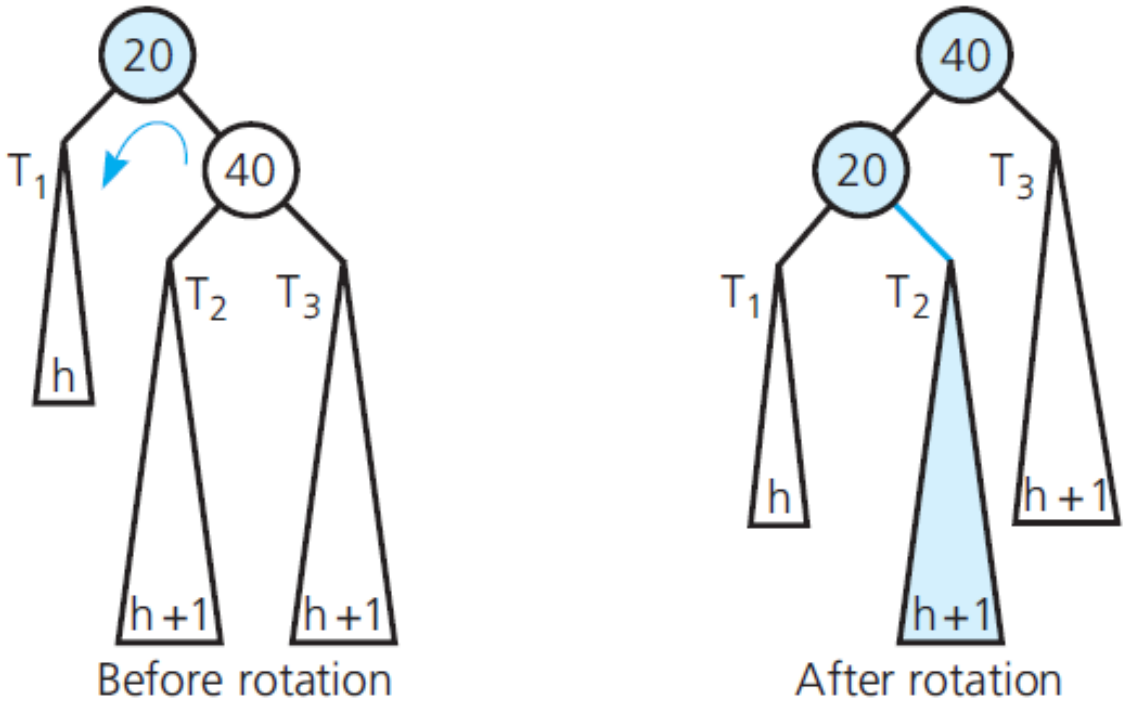
A single rotation to the left that does not affect the height of an AVL tree



AVL Trees

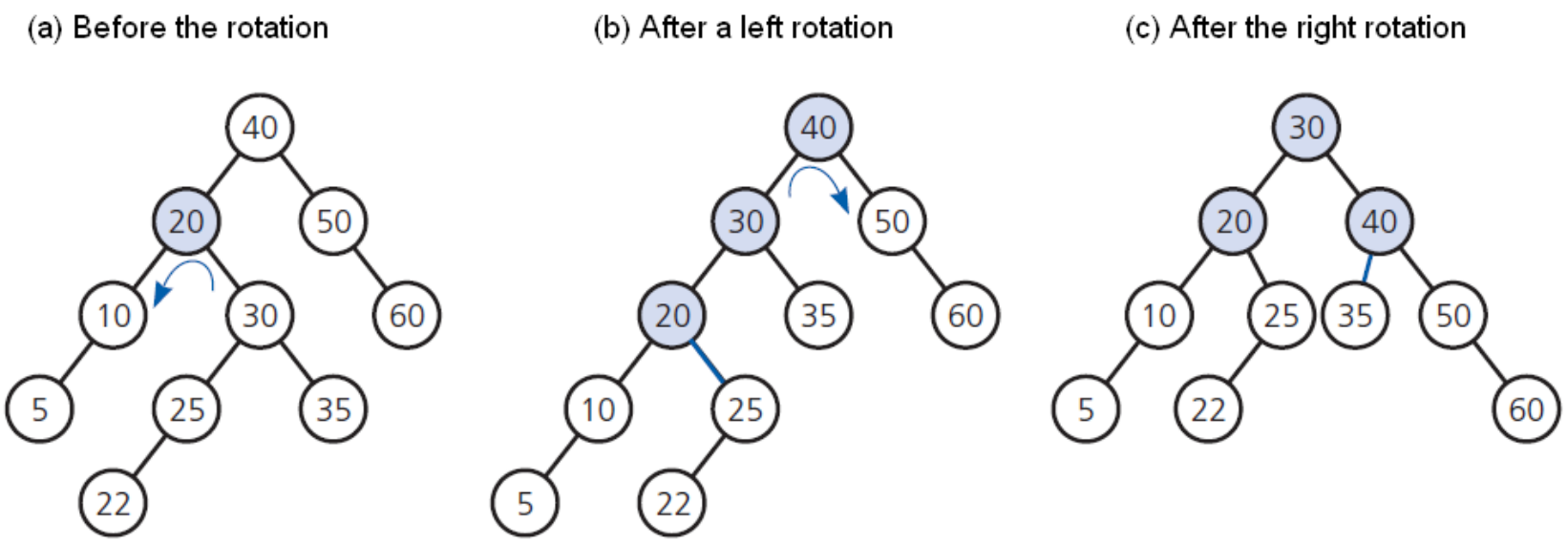
[Continued]

(c) The general case for a single left rotation in an AVL tree whose height is unchanged



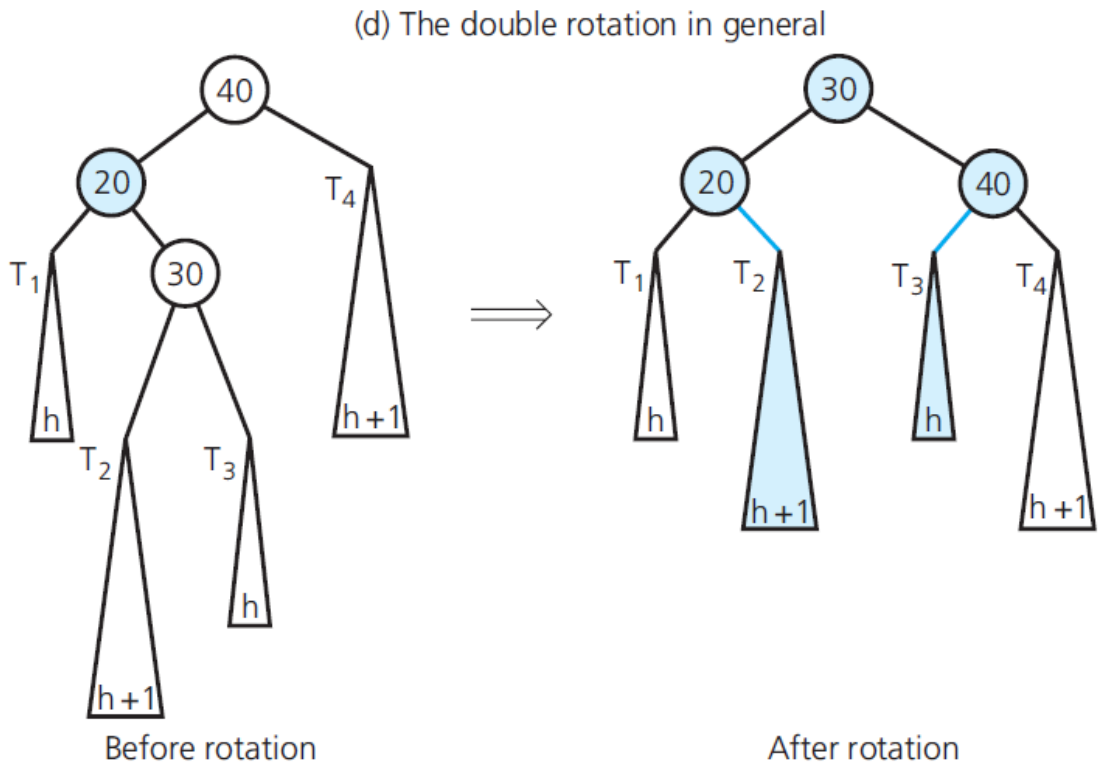
AVL Trees

A double rotation that decreases the height of an AVL tree



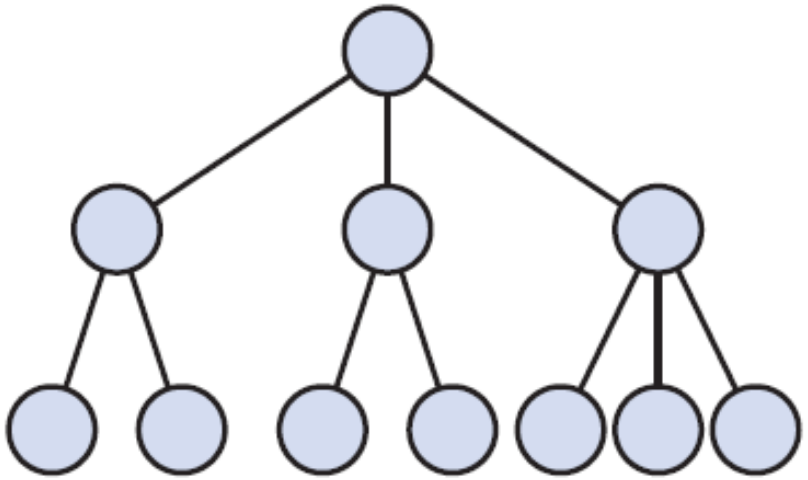
AVL Trees

[Continued]



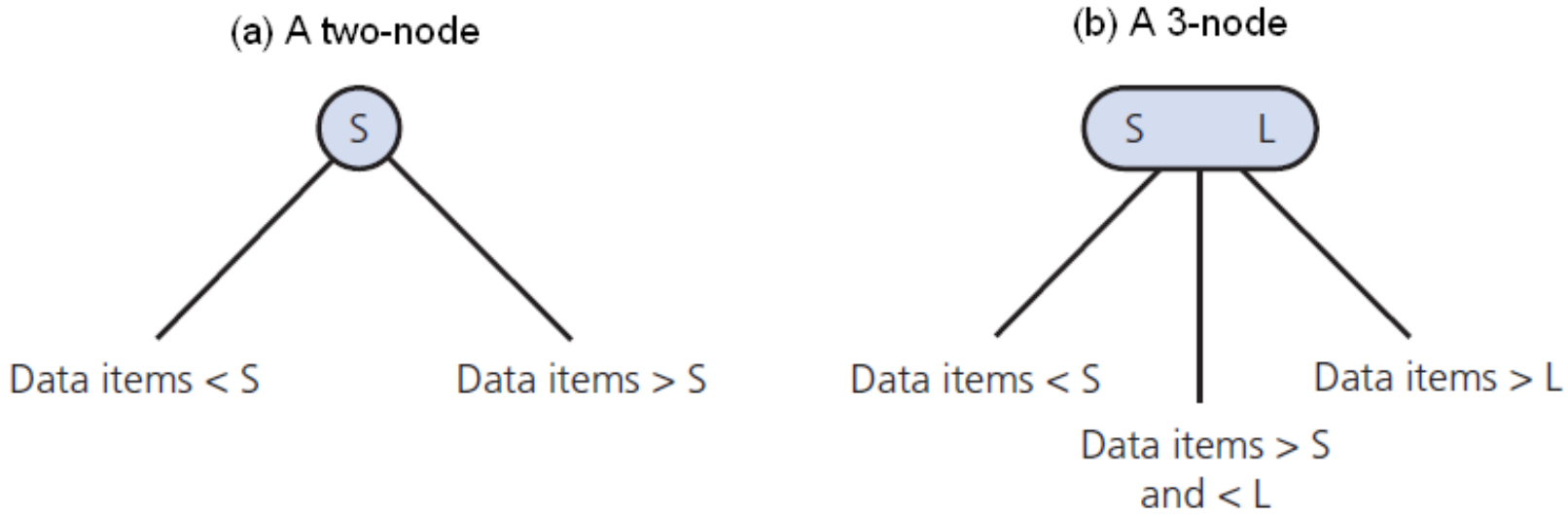
2-3 Trees

A 2-3 tree of height 3



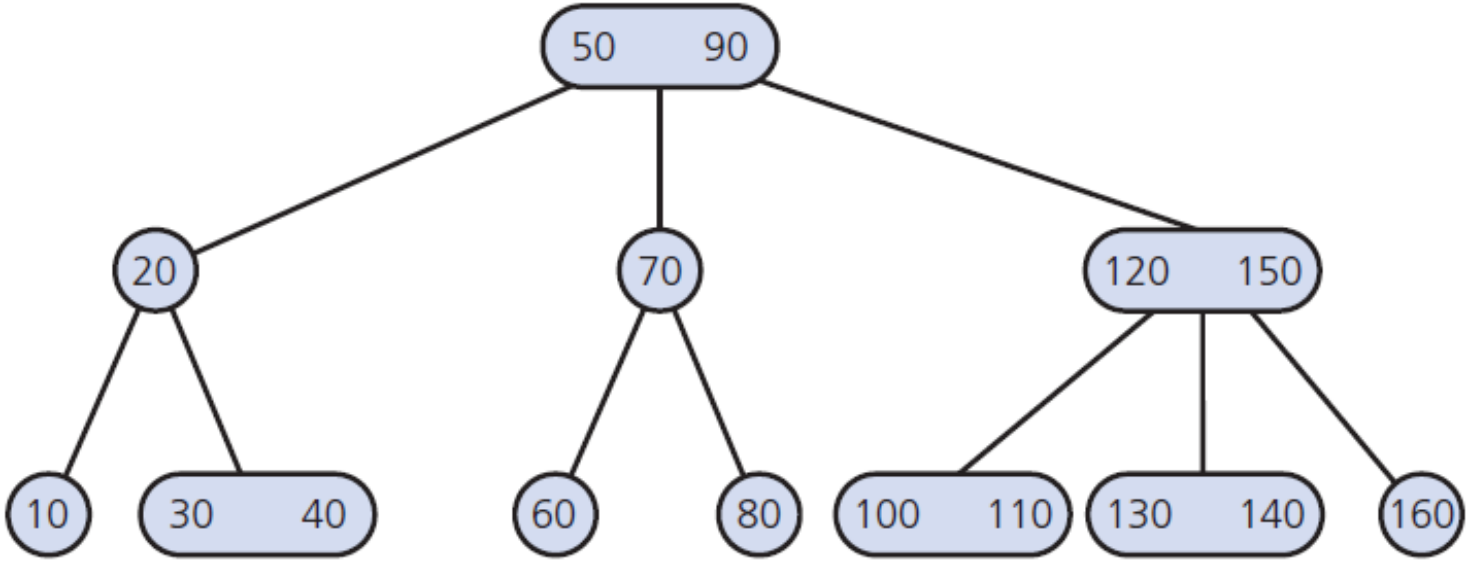
2-3 Trees

Nodes in a 2-3 tree



2-3 Trees

A 2-3 tree



2-3 Trees

A header file for a class of nodes for a 2-3 tree

```
1  /** A class of nodes for a link-based 2-3 tree.
2   @file TriNode.h */
3
4  #ifndef TRI_NODE_
5  #define TRI_NODE_
6
7  template<class ItemType>
8  class TriNode
9  {
10 private:
11    ItemType smallItem;                                  // Data portion
12    ItemType largeItem;                                  // Data portion
13    std::shared_ptr<TriNode<ItemType>> leftChildPtr;   // Left-child pointer
14    std::shared_ptr<TriNode<ItemType>> midChildPtr;    // Middle-child pointer
15    std::shared_ptr<TriNode<ItemType>> rightChildPtr; // Right-child pointer
16
17 public:
18    TriNode();
```

2-3 Trees

[Continued]

```
19
20      bool isLeaf() const;
21      bool isTwoNode() const;
22      bool isThreeNode() const;
23
24      ItemType getSmallItem() const;
25      ItemType getLargeItem() const;
26
27      void setSmallItem(const ItemType& anItem);
28      void setLargeItem(const ItemType& anItem);
29      auto getLeftChildPtr() const;
30      auto getMidChildPtr() const;
31      auto getRightChildPtr() const;
32
33      void setLeftChildPtr(std::shared_ptr<TriNode<ItemType>> leftPtr);
34      void setMidChildPtr(std::shared_ptr<TriNode<ItemType>> midPtr);
35      void setRightChildPtr(std::shared_ptr<TriNode<ItemType>> rightPtr);
36  }; // end TriNode
37  #include “TriNode.cpp”
38  #endif
```

Traversing a 2-3 Tree

Performing the analogue of an inorder traversal on a binary tree:

```
// Traverses a nonempty 2-3 tree in sorted order.
inorder(23Tree: TwoThreeTree): void
{
   if (23Tree's root node r is a leaf)
      Visit the data item(s)
   else if (r has two data items)
   {
      inorder(left subtree of 23Tree's root)
      Visit the first data item
      inorder(middle subtree of 23Tree's root)
      Visit the second data item
      inorder(right subtree of 23Tree's root)
   }
   else // r has one data item
   {
      inorder(left subtree of 23Tree's root)
      Visit the data item
      inorder(right subtree of 23Tree's root)
   }
}
```

Searching a 2-3 Tree

Retrieval operation for a 2-3 tree

```
// Locates the value target in a nonempty 2-3 tree. Returns either the located
// entry or throws an exception if such a node is not found.
findItem(23Tree: TwoThreeTree, target: ItemType): ItemType
{
   if (target is in 23Tree's root node r)
   {  // The data item has been found
      treeItem = the data portion of r
      return treeItem // Success
   }
   else if (r is a leaf)
      throw NotFoundException   // Failure

   // Else search the appropriate subtree
   else if (r has two data items)
   {
      if (target < smaller data item in r)
```

Searching a 2-3 Tree

Retrieval operation for a 2-3 tree

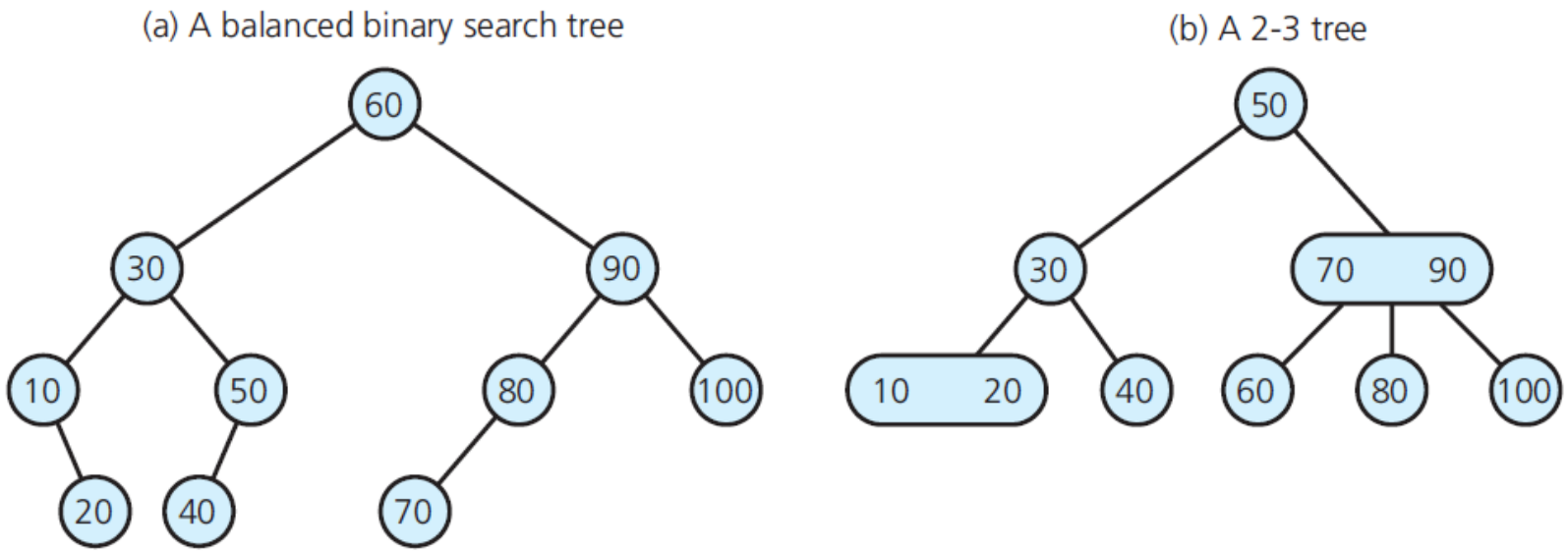
```
    else if (r has two data items)
    {
       if (target < smaller data item in r)
          return findItem(r's left subtree, target)
       else if (target < larger data item in r)
          return findItem(r's middle subtree, target)
       else
          return findItem(r's right subtree, target)
    }
    else // r has one data item
    {
       if (target < r's data item)
          return findItem(r's left subtree, target)
       else
          return findItem(r's right subtree, target)
    }
}
```

Searching a 2-3 Tree

- Search of a 2-3 and shortest binary search tree approximately same efficiency
  - A binary search tree with *n* nodes cannot be shorter than $\log_2(n + 1)$
  - A 2-3 tree with n nodes cannot be taller than $\log_2(n + 1)$
  - Node in a 2-3 tree has at most two data items
- Searching 2-3 tree is $O(\log n)$

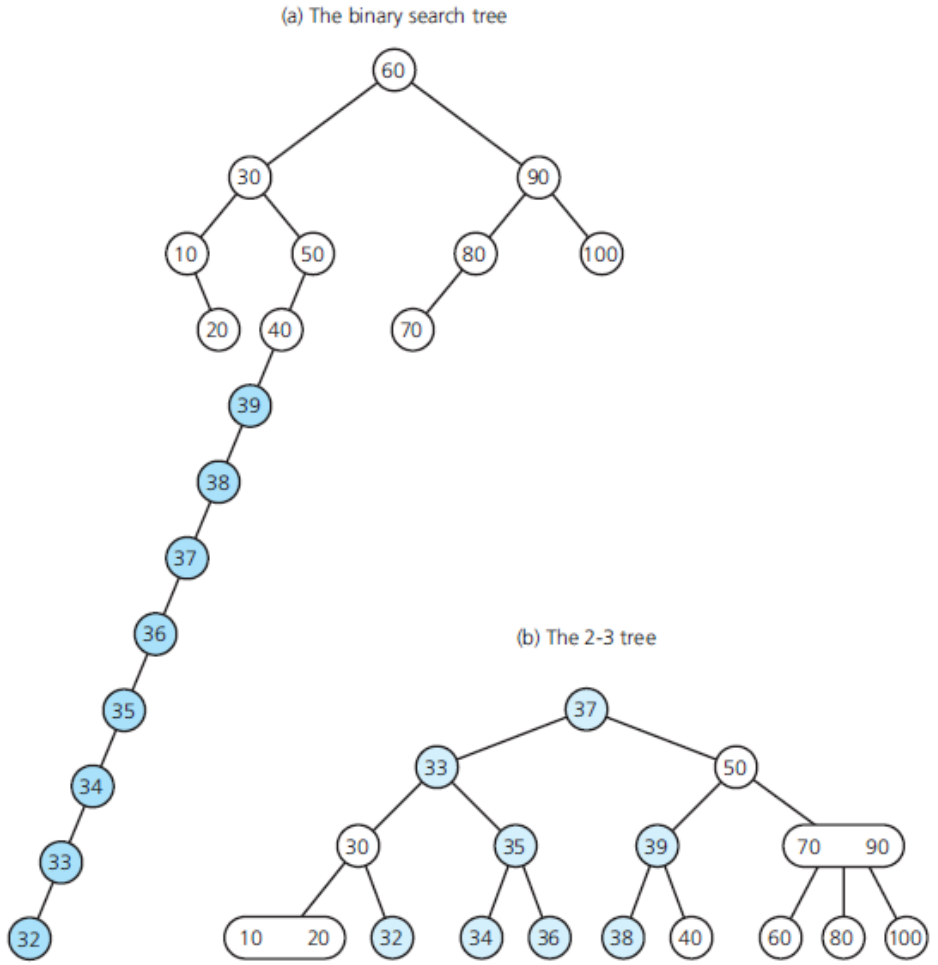
Searching a 2-3 Tree

A balanced binary search tree & a 2-3 tree



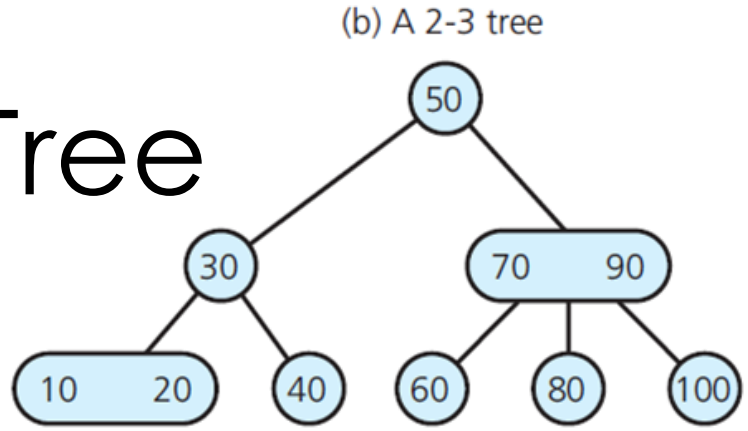
Searching a 2-3 Tree

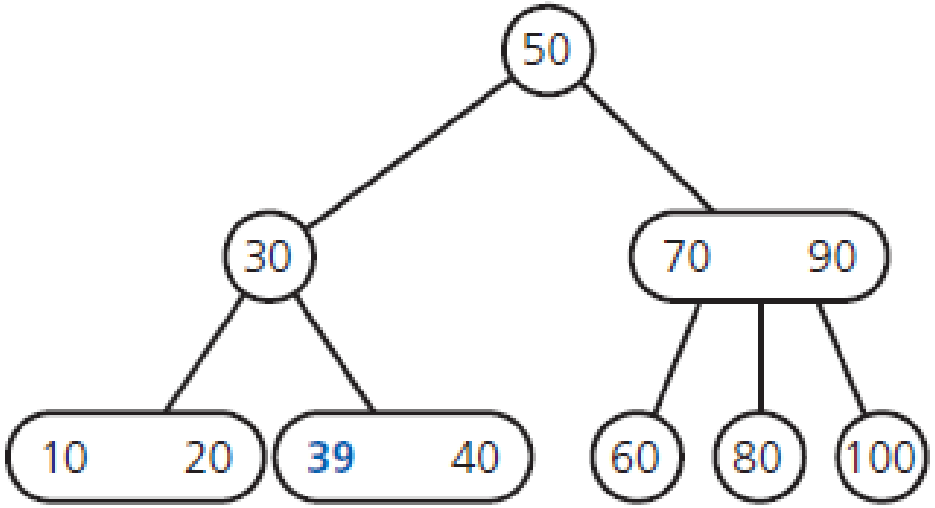
The trees shown before after adding the values 39 down to 32



Adding Data to a 2-3 Tree

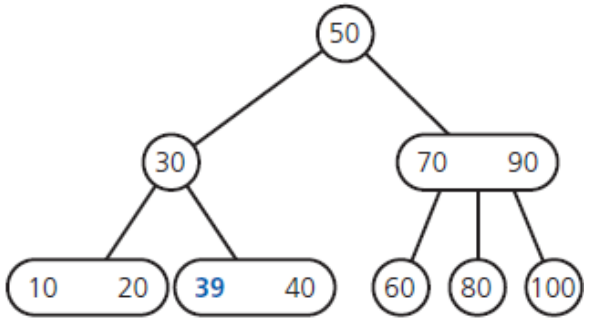
After inserting 39 into the tree in previous slide



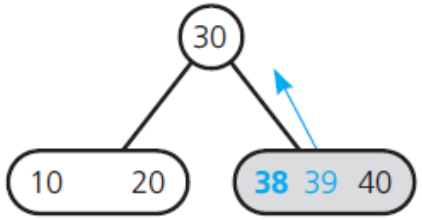


Adding Data to a 2-3 Tree

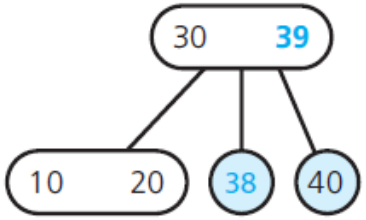
The steps for adding 38 to the tree in previous figure



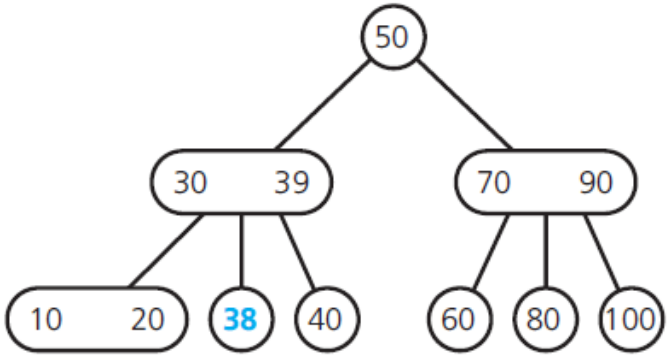
(a) The located node has no room



(b) The node splits and 39 moves up

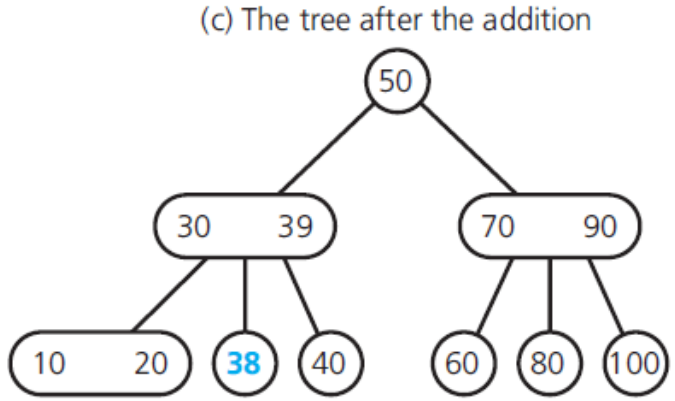


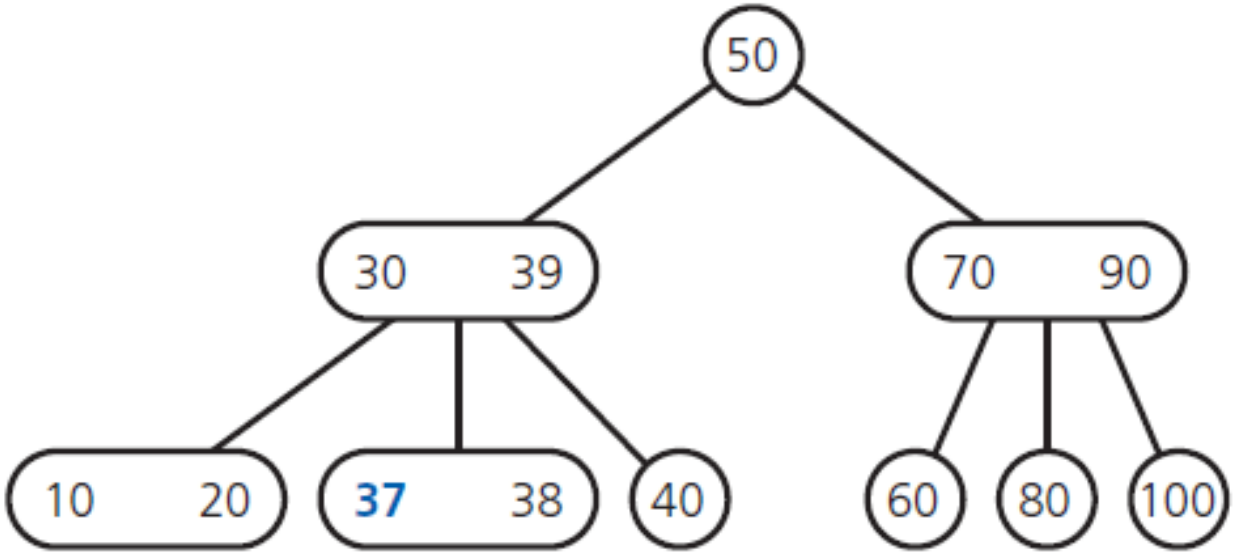
(c) The tree after the addition



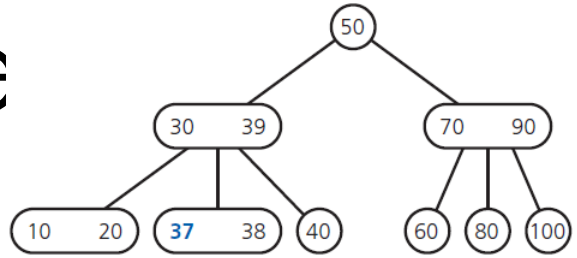
Adding Data to a 2-3 Tree

After adding 37 to the tree in Figure

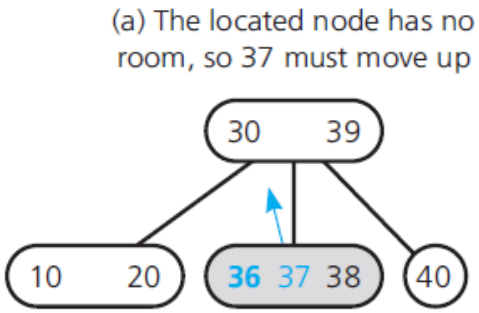




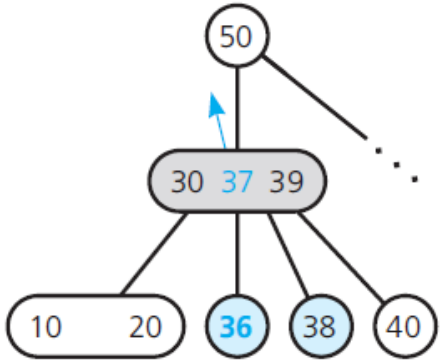
Adding Data to a 2-3 Tree



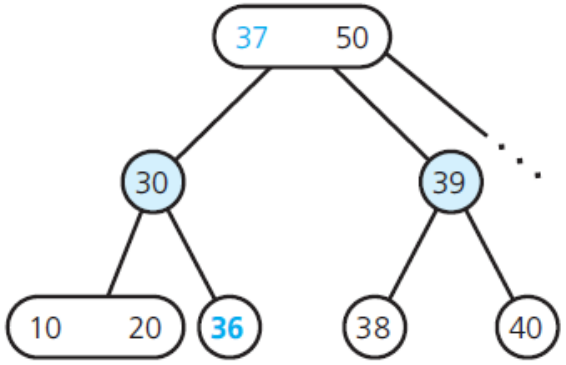
The steps for adding 36 to the tree in Figure



(a) The located node has no room, so 37 must move up



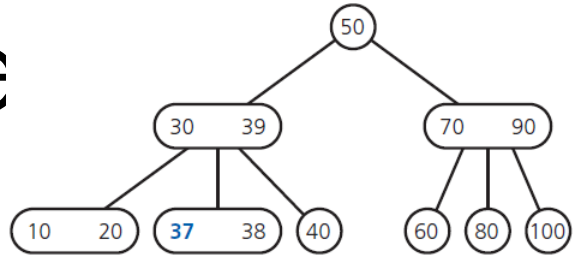
(b) After the node splits, its parent has no room for 37



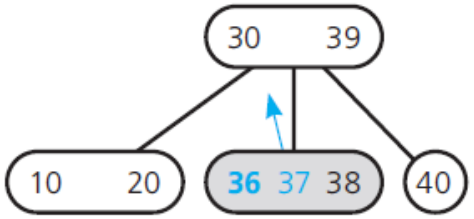
(c) The node splits and 37 moves up

Adding Data to a 2-3 Tree

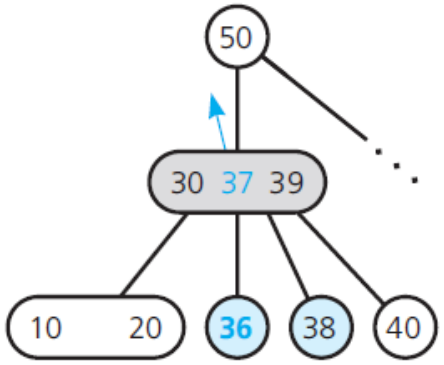
The steps for adding 36 to the tree in Figure



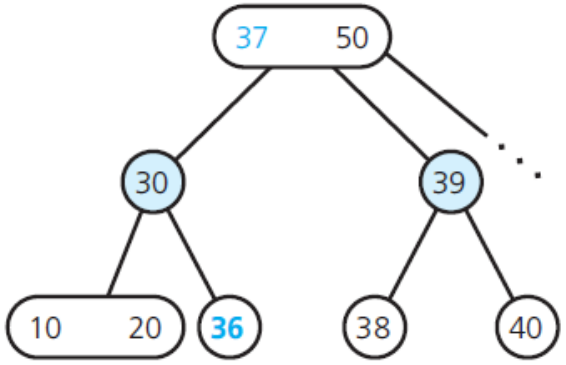
(a) The located node has no room, so 37 must move up



(b) After the node splits, its parent has no room for 37



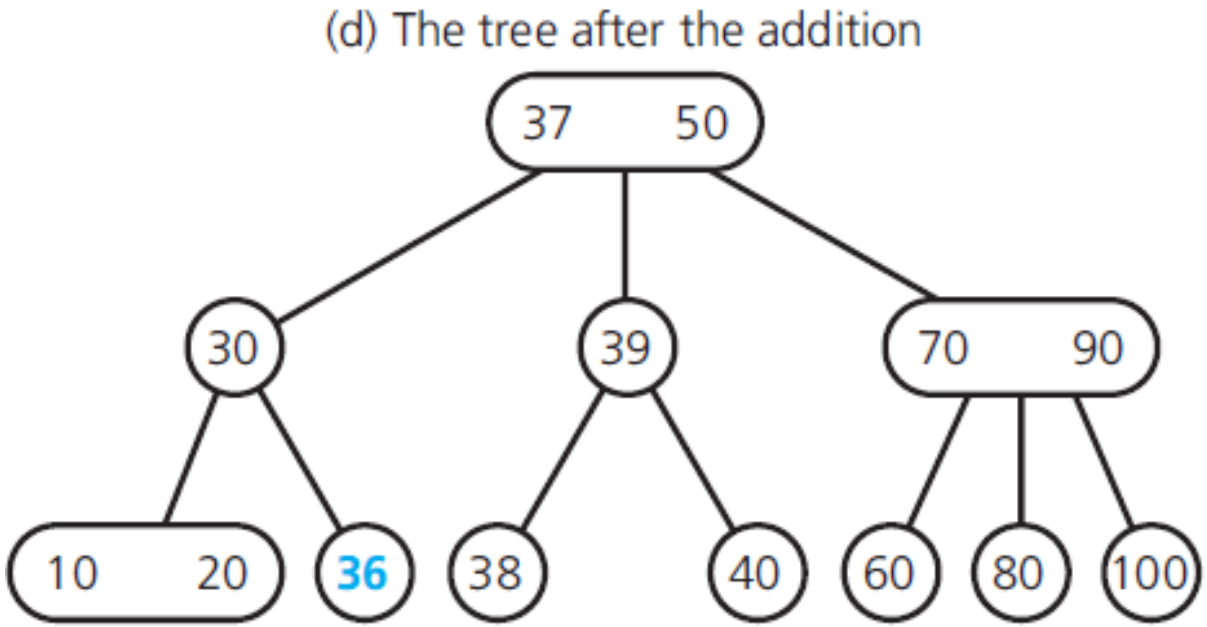
(c) The node splits and 37 moves up



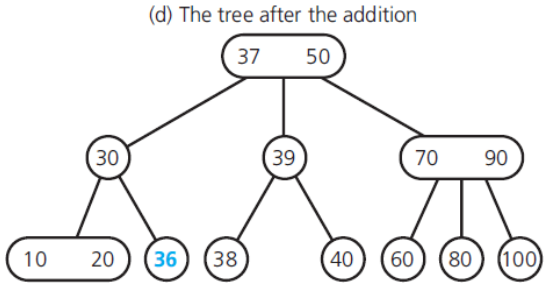
bubbles up

Adding Data to a 2-3 Tree

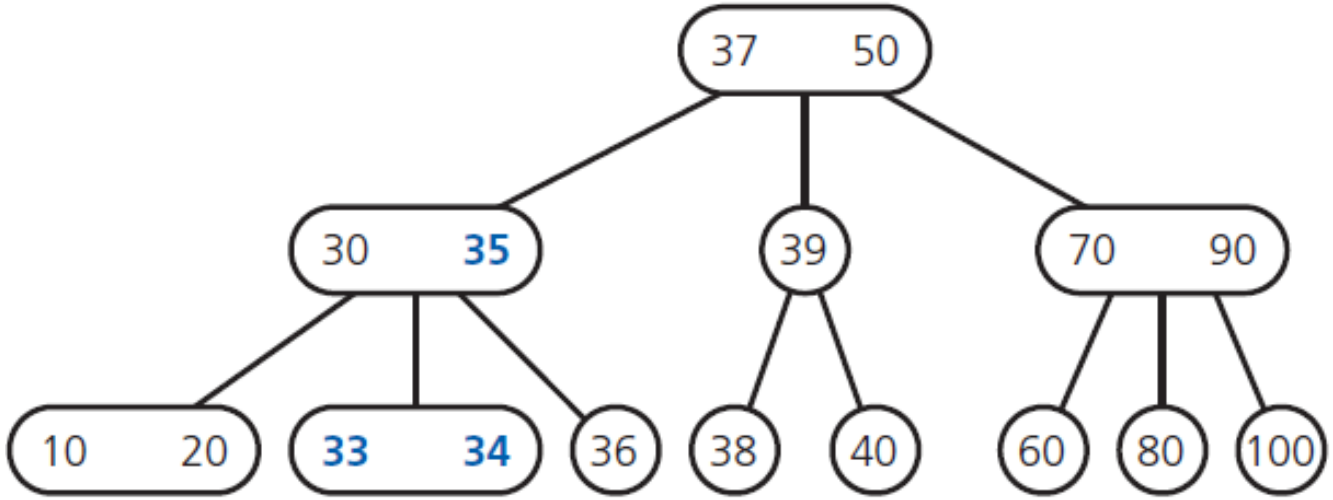
[Continued]



Adding Data to a 2-3 Tree (6 of 11)

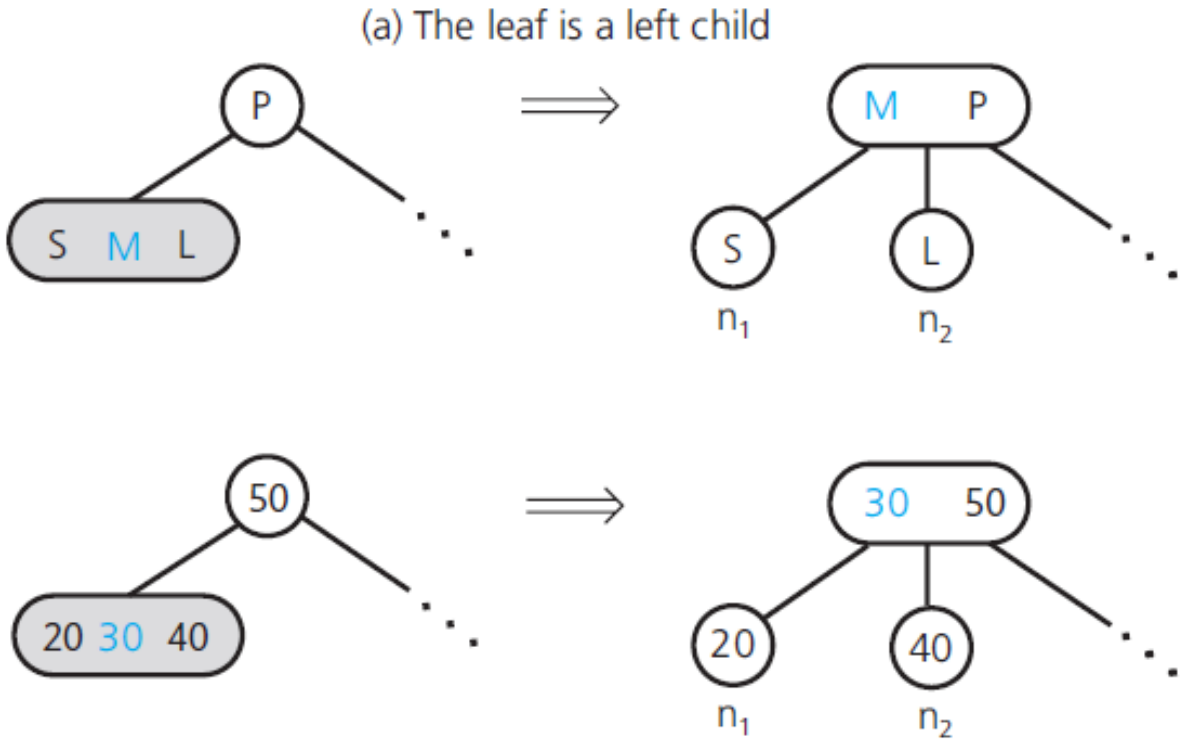


The tree after the adding 35, 34, and 33 to the tree in Figure



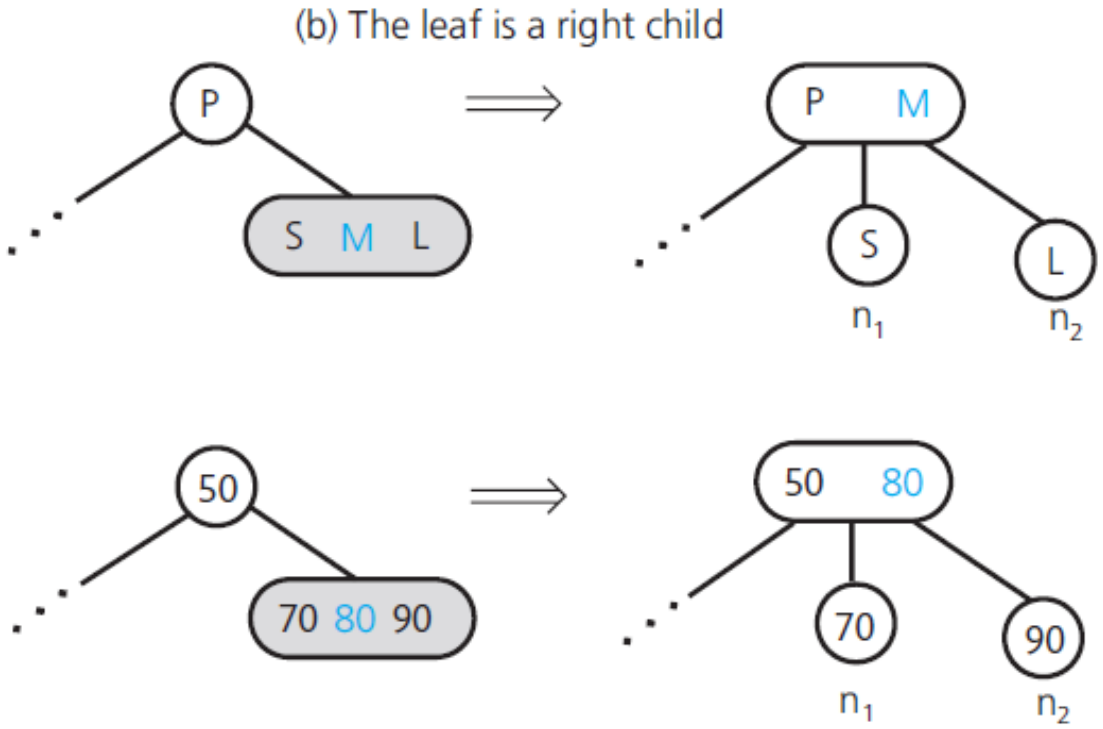
Adding Data to a 2-3 Tree

Splitting a leaf in a 2-3 tree in general and in a specific example



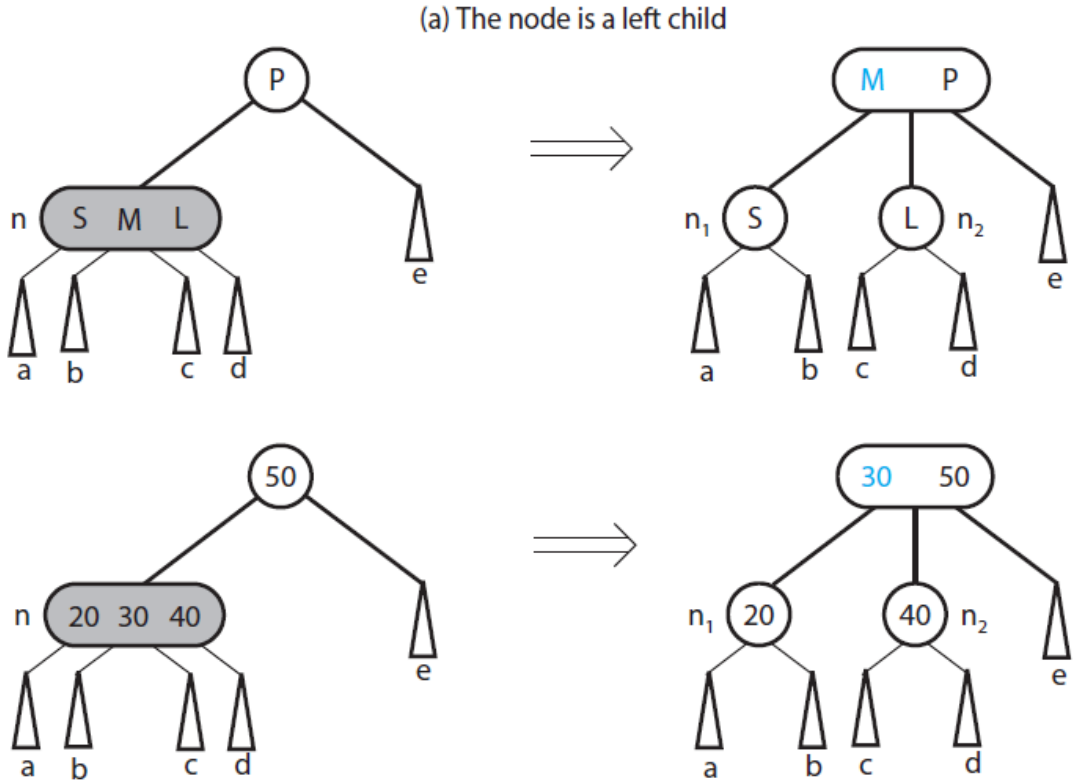
Adding Data to a 2-3 Tree

[Continued]



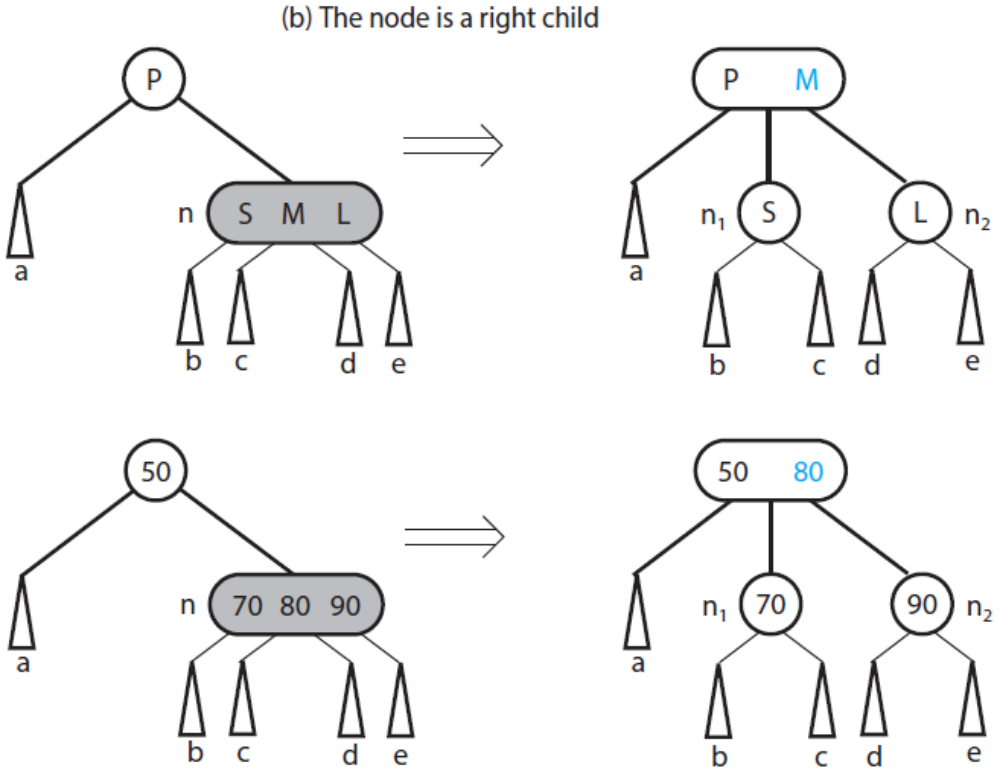
Adding Data to a 2-3 Tree

Splitting an internal node in a 2-3 tree in general and in a specific example



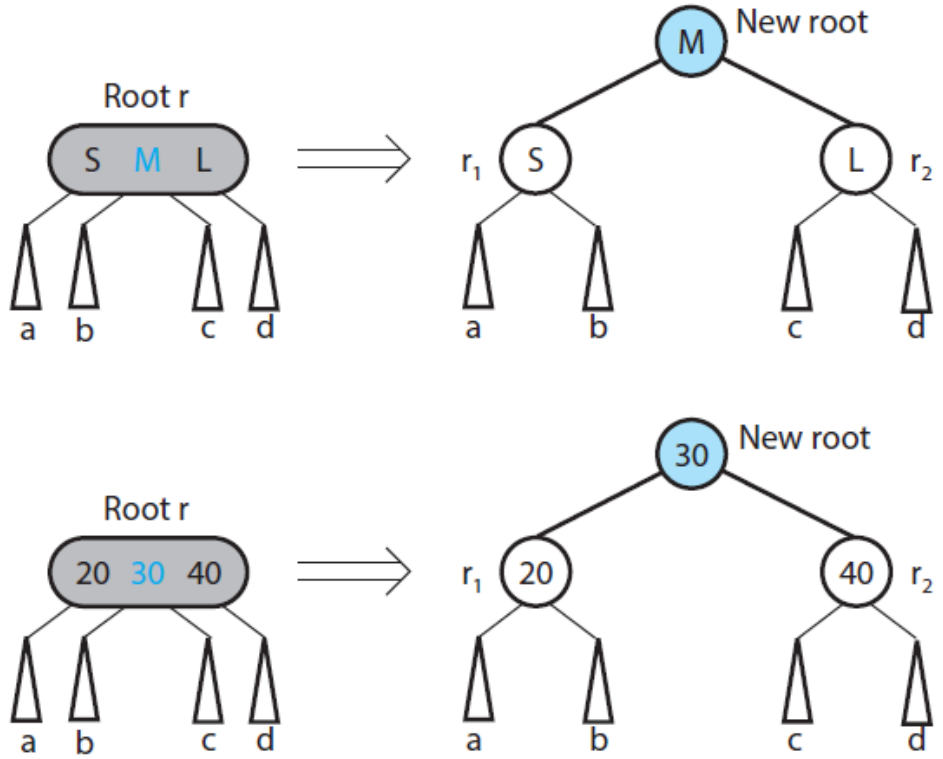
Adding Data to a 2-3 Tree

[Continued]



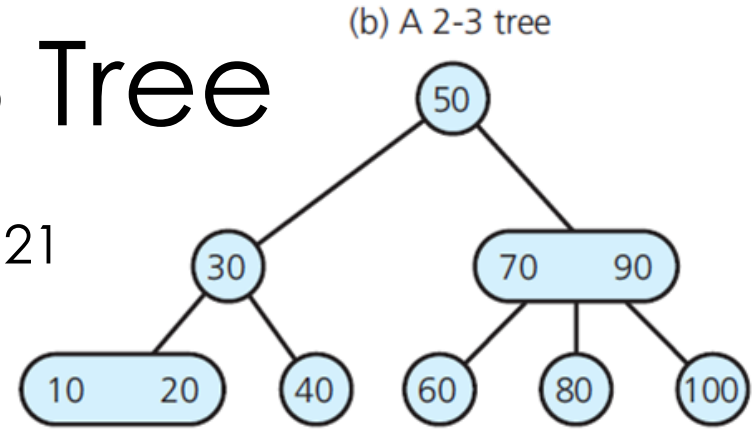
Adding Data to a 2-3 Tree

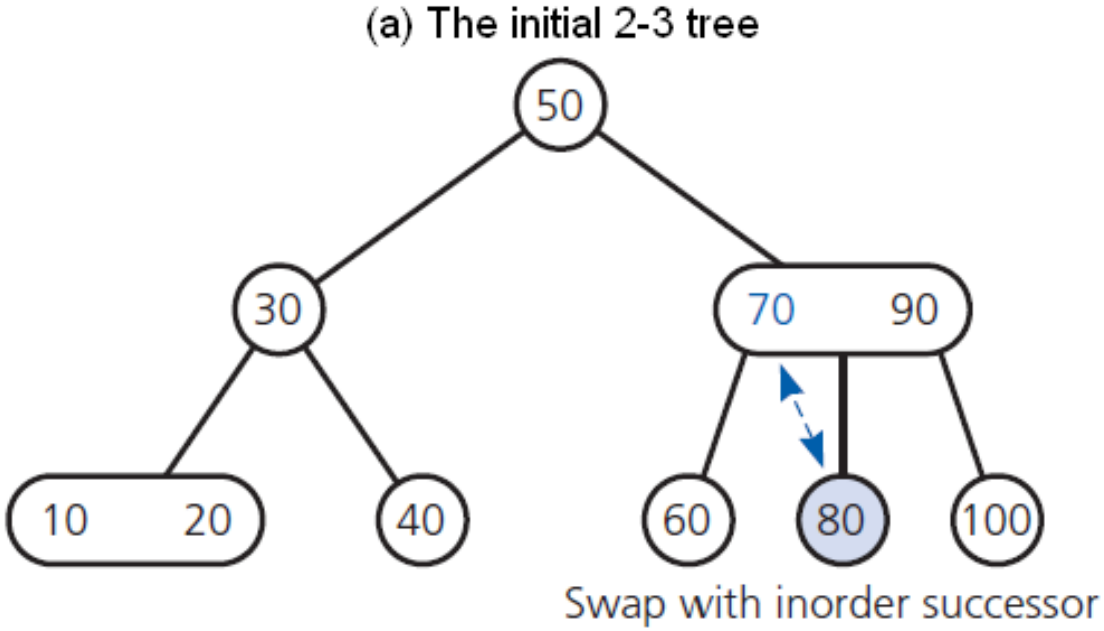
Splitting the root of a 2-3 tree general and in a specific example



Removing Data from a 2-3 Tree

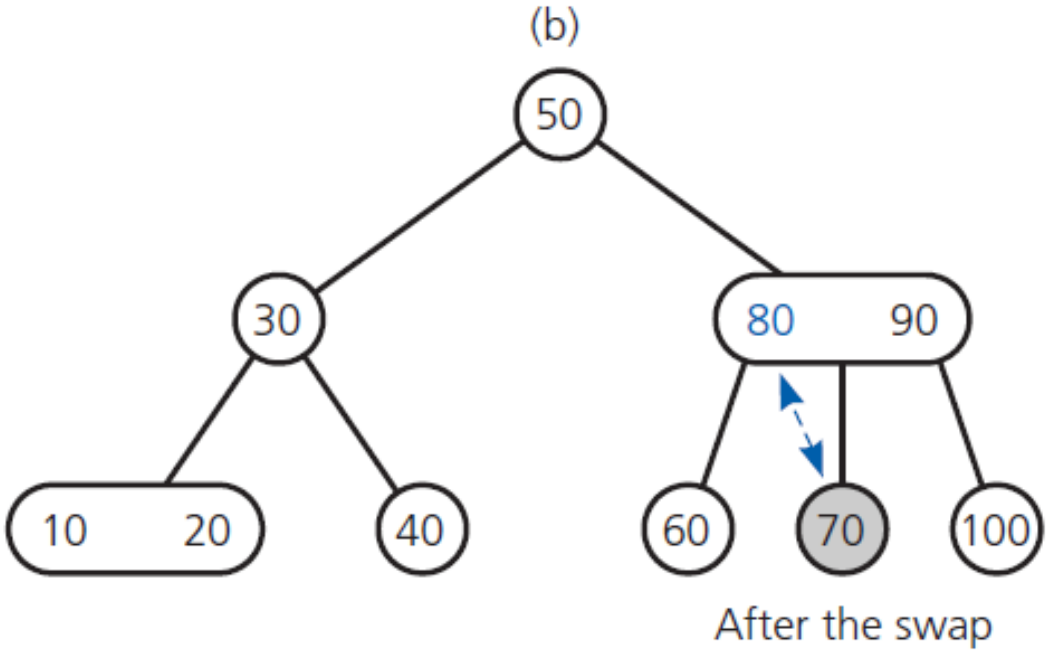
The steps for removing 70 from the 2-3 tree in Figure - slide 21





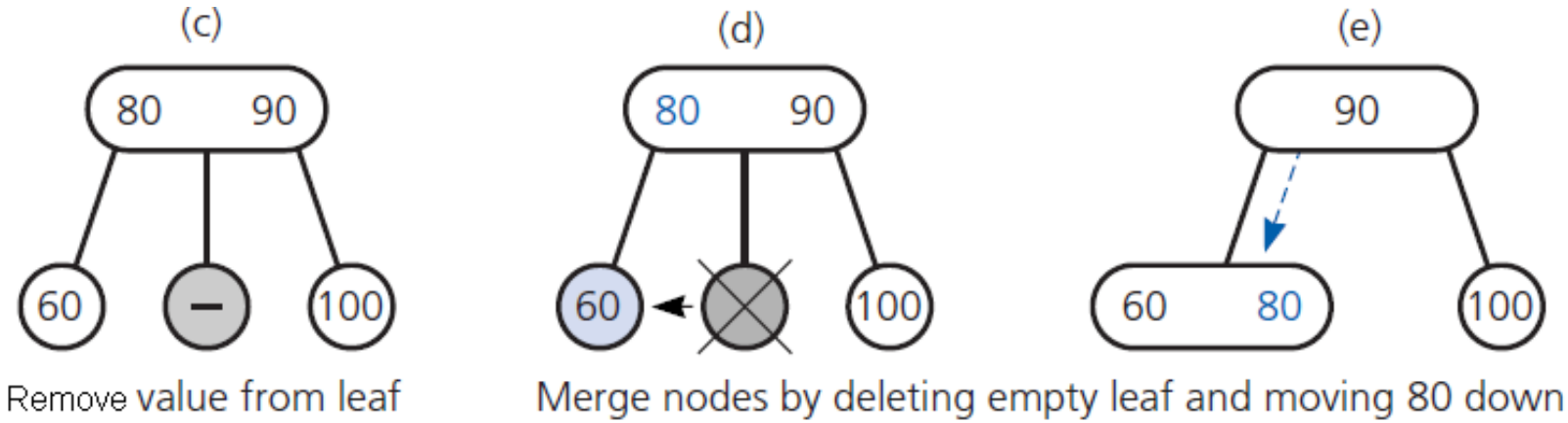
Removing Data from a 2-3 Tree

[Continued]



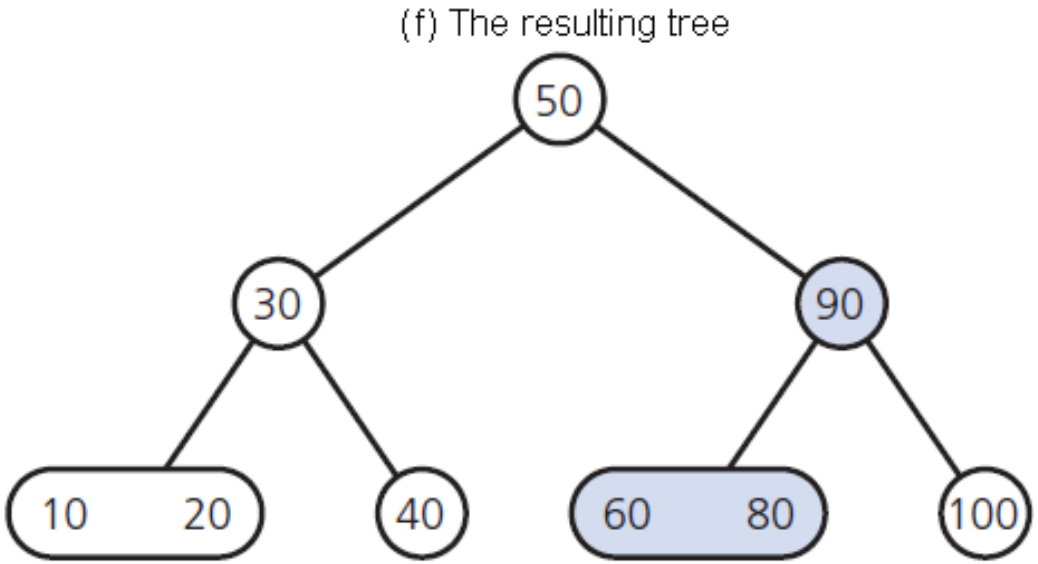
Removing Data from a 2-3 Tree

[Continued]



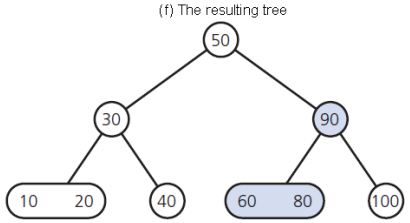
Removing Data from a 2-3 Tree

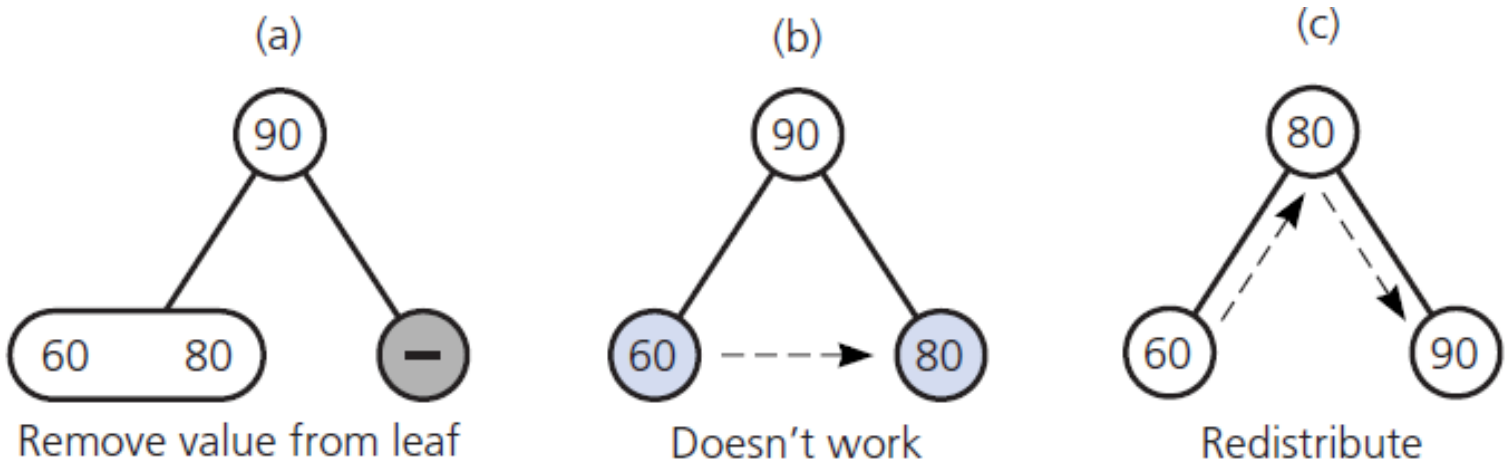
[Continued]



Removing Data from a 2-3 Tree

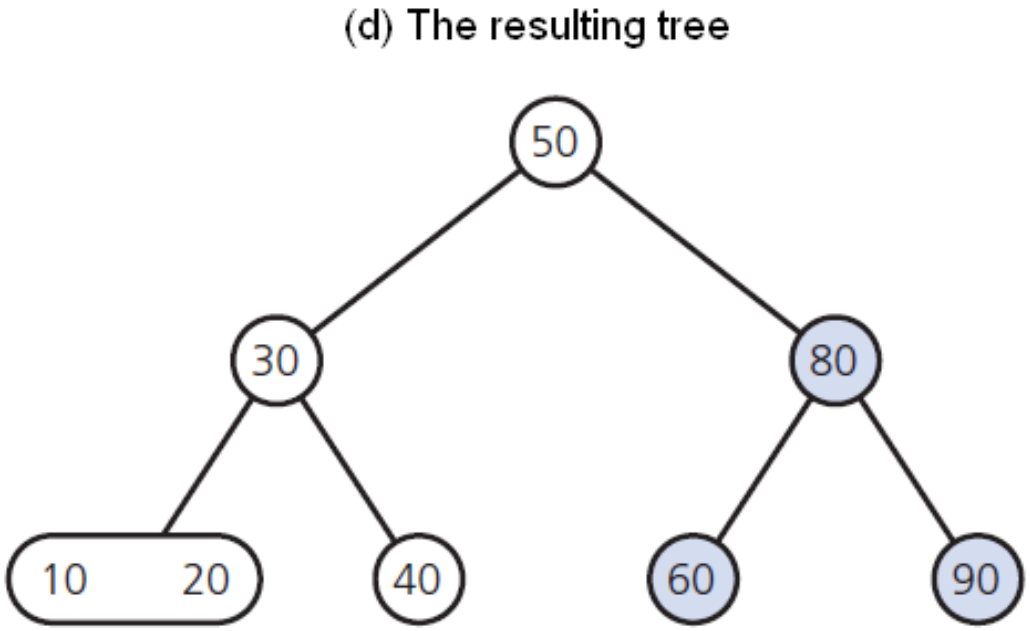
The steps for removing 100 from the tree in Figure





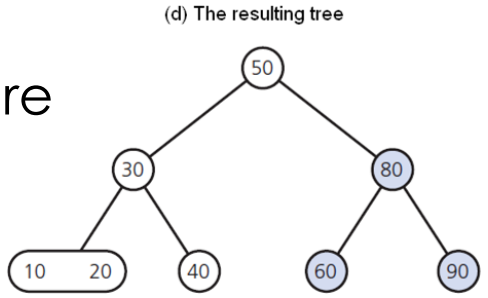
Removing Data from a 2-3 Tree

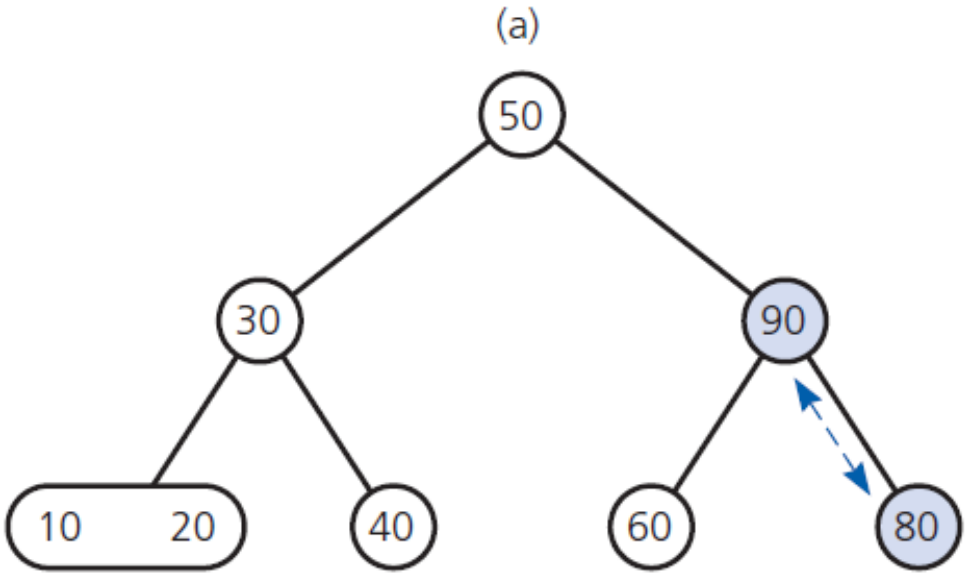
[Continued]



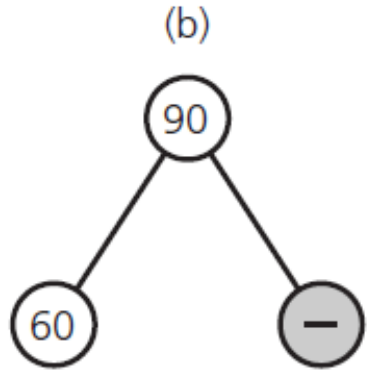
Removing Data from a 2-3 Tree

The steps for removing 80 from the 2-3 tree in Figure





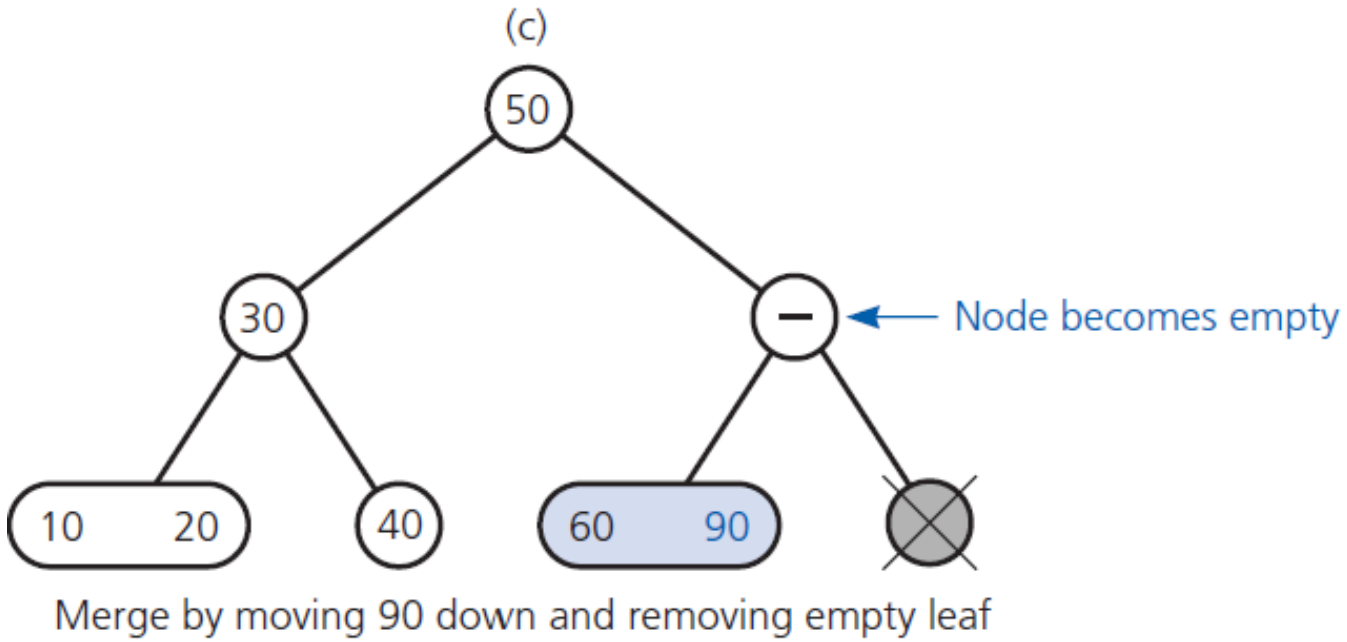
After swap with inorder successor



Remove value from leaf

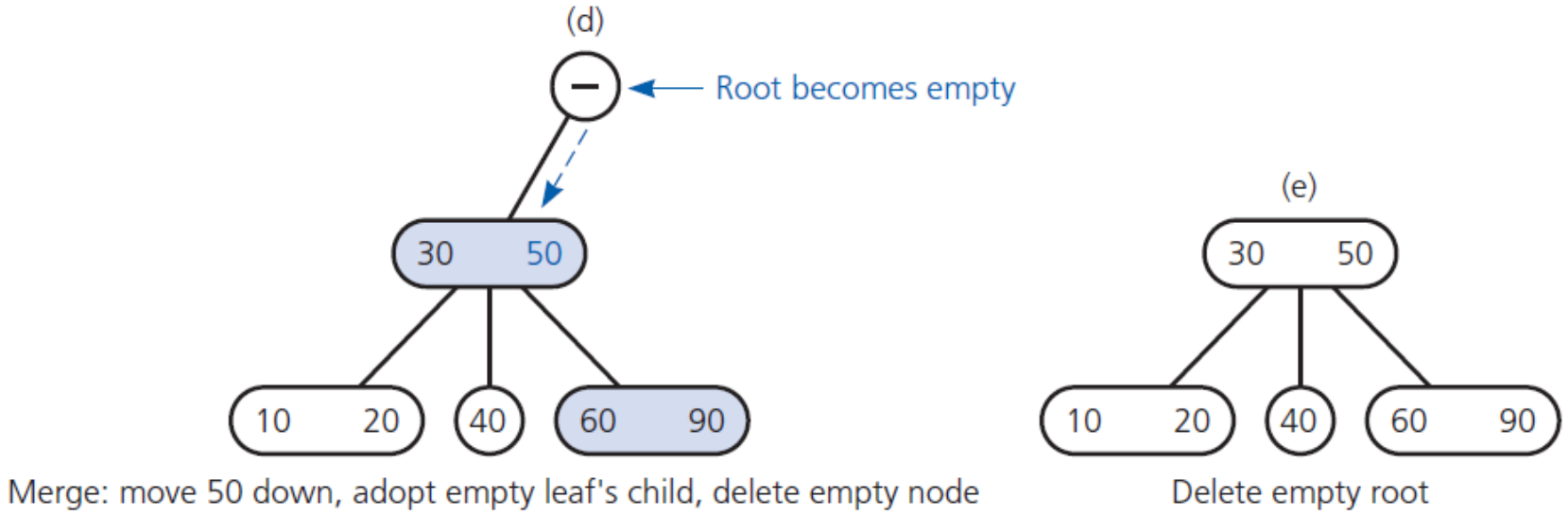
Removing Data from a 2-3 Tree

[Continued]



Removing Data from a 2-3 Tree

[Continued]

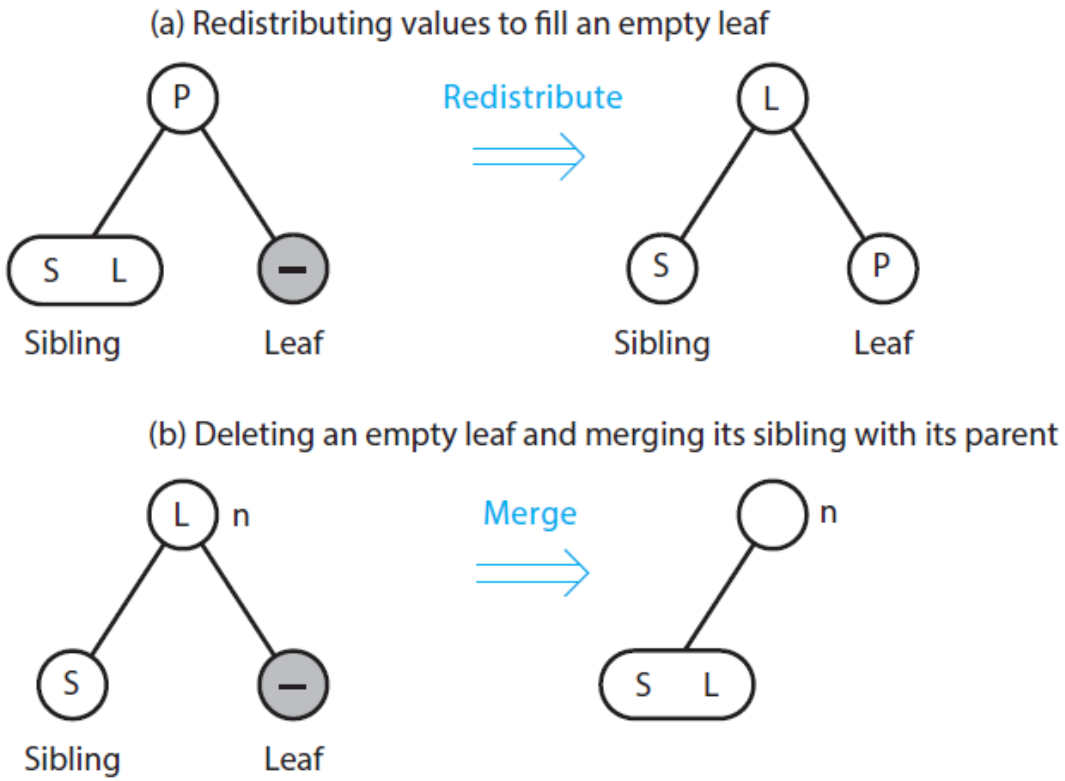


Merge: move 50 down, adopt empty leaf's child, delete empty node

Delete empty root

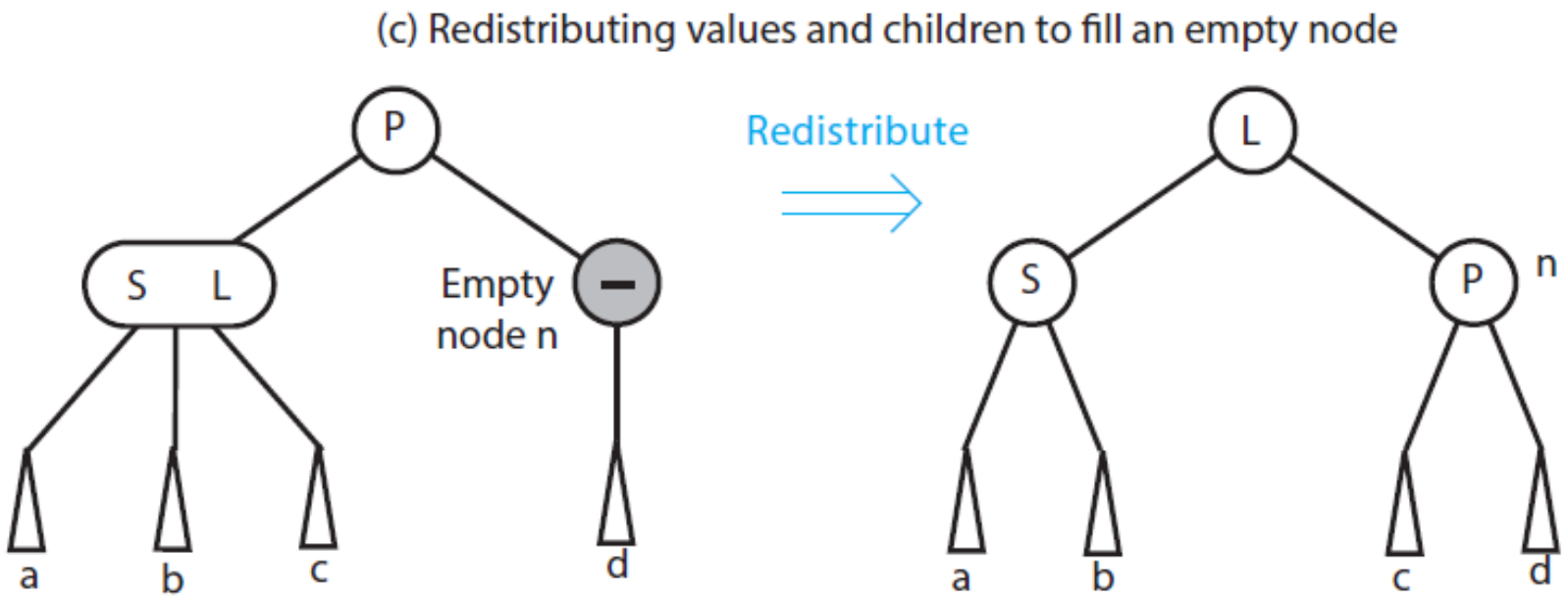
Removing Data from a 2-3 Tree

Possible situations during the removal of a data item from a 2-3 tree



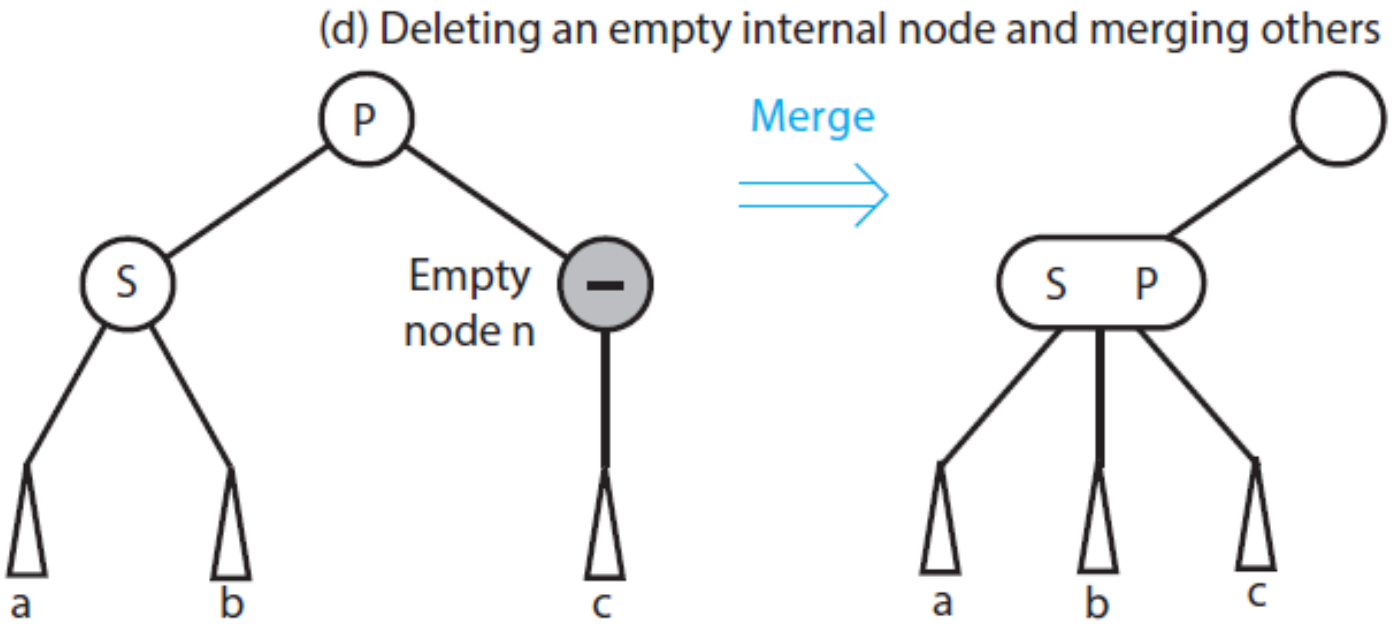
Removing Data from a 2-3 Tree

[Continued]



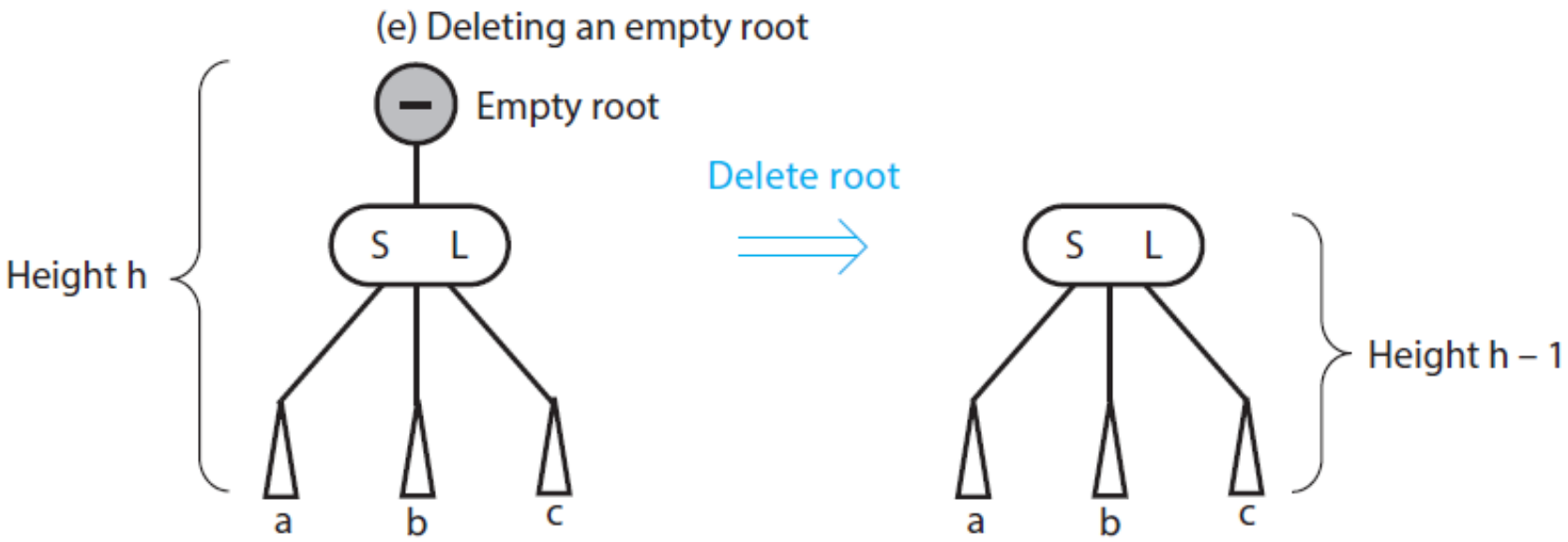
Removing Data from a 2-3 Tree

[Continued]



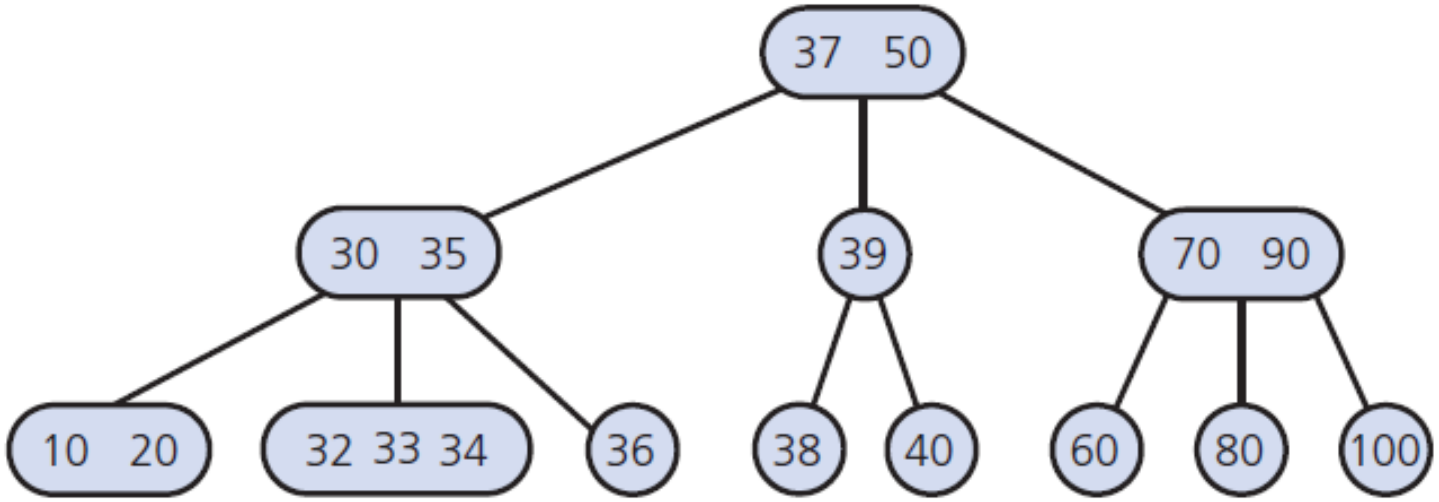
Removing Data from a 2-3 Tree

[Continued]



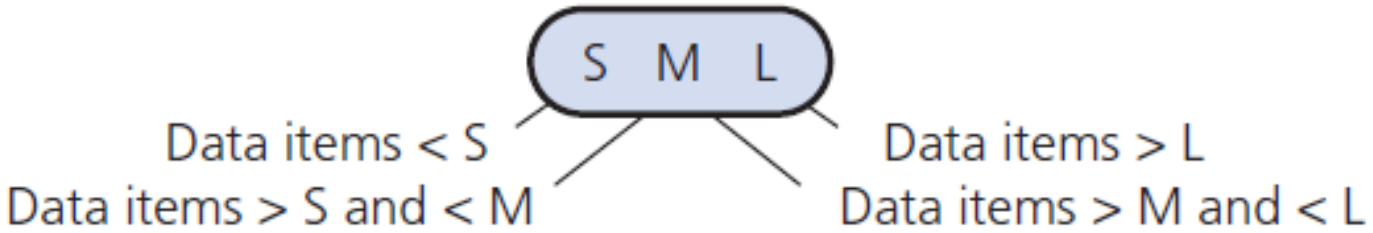
2-3-4 Trees

A 2-3-4 tree with the same data items as the 2-3 tree in Figure - slide 22



2-3-4 Trees

A 4-node in a 2-3-4 tree

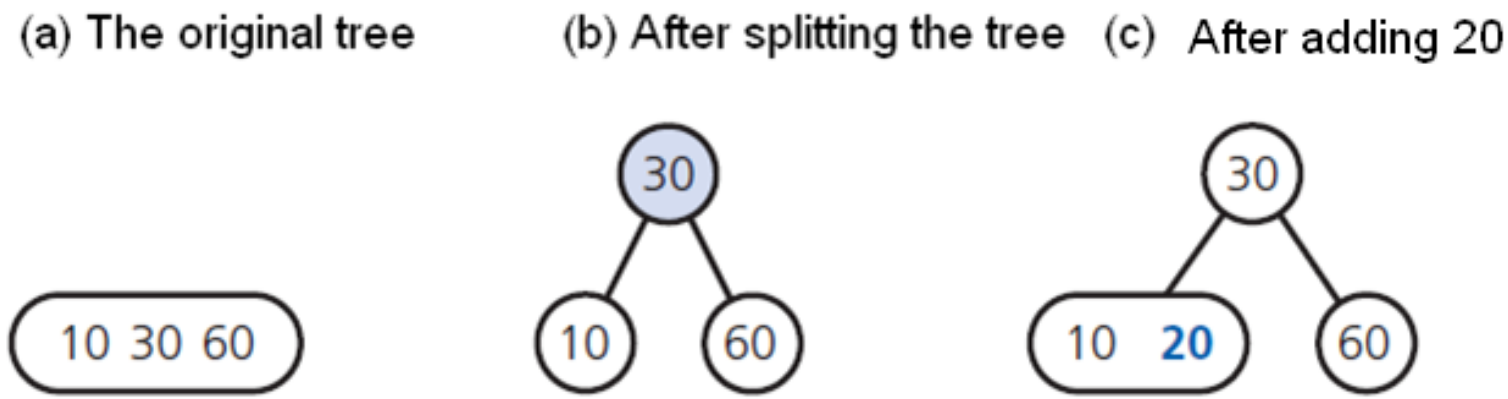


2-3-4 Trees

- Searching and traversing
  - Simple extensions of corresponding algorithms for a 2-3 tree
- Adding data
  - Like addition algorithm for 2-3 tree
  - Splits node by moving one data item up to parent node

Adding Data to 2-3-4 Trees

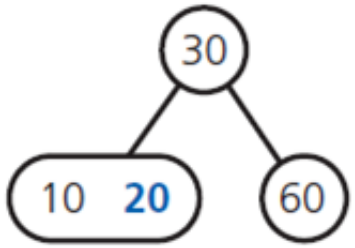
Adding 20 to a one-node 2-3-4 tree

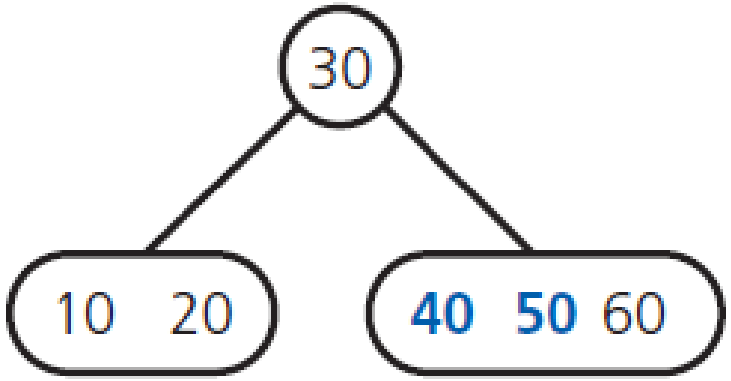


Adding Data to 2-3-4 Trees

After adding 50 and 40 to the tree in Figure

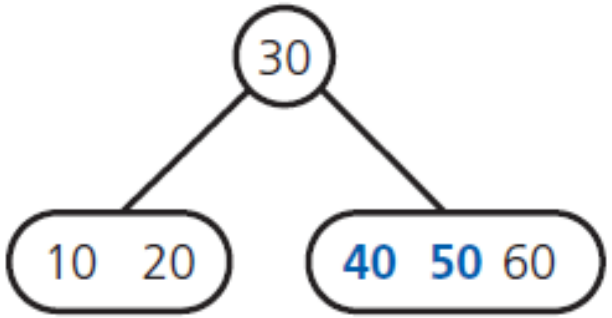
(c) After adding 20



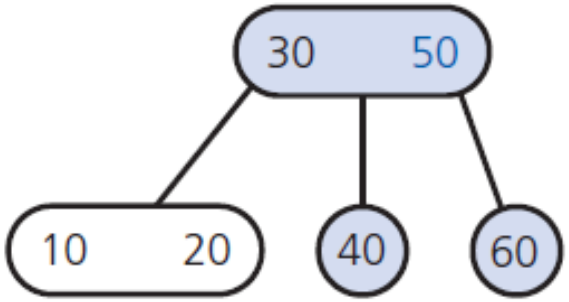


Adding Data to 2-3-4 Trees

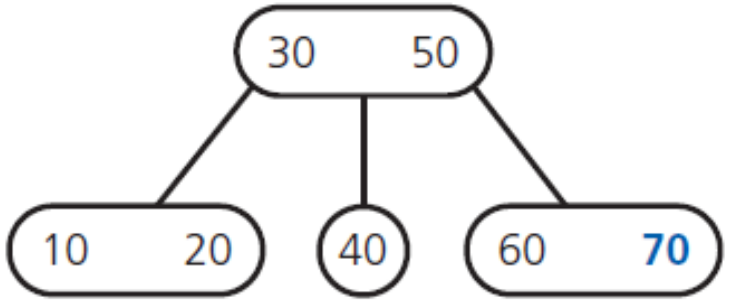
The steps for adding 70 to the tree in Figure



(a) After splitting the 4-node

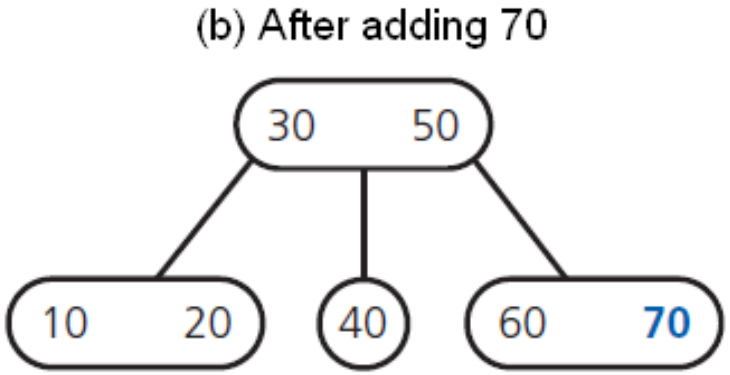


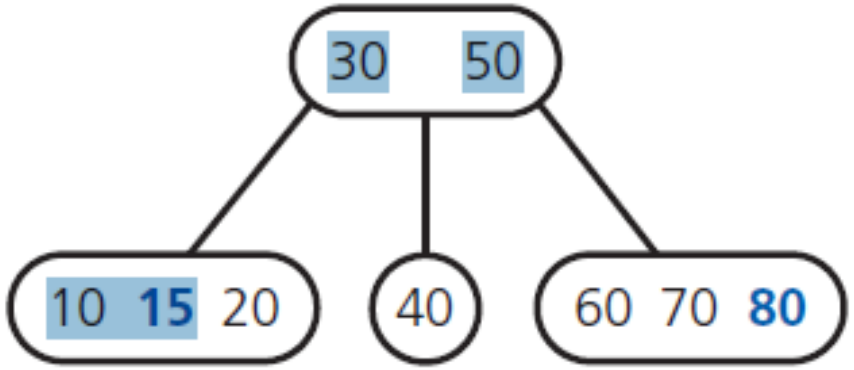
(b) After adding 70



Adding Data to 2-3-4 Trees

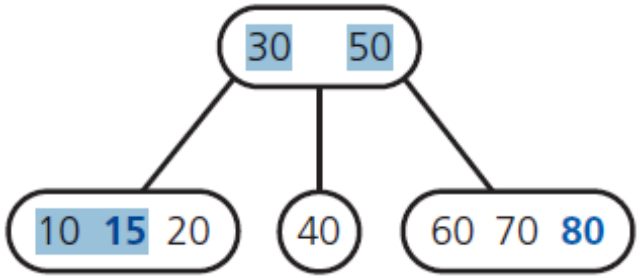
After adding 80 and 15 to the tree in Figure



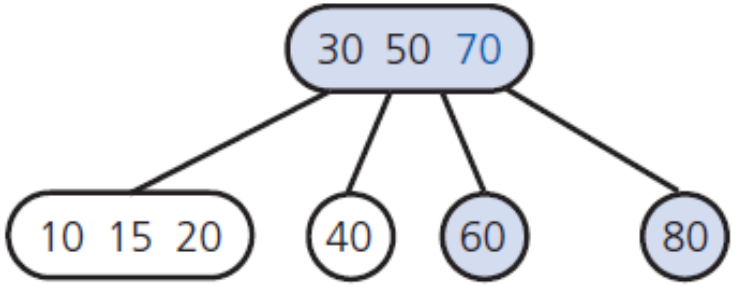


Adding Data to 2-3-4 Trees

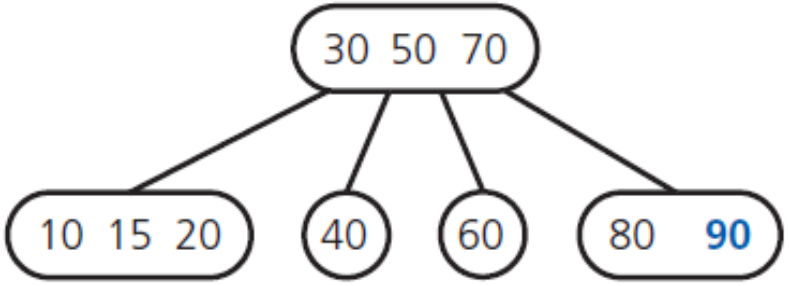
The steps for adding 90 to the tree in Figure



(a) After splitting the root's right child



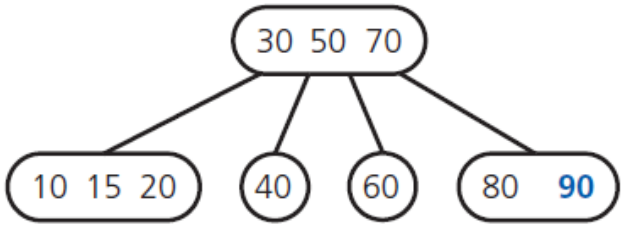
(b) After adding 90 to the root's right child



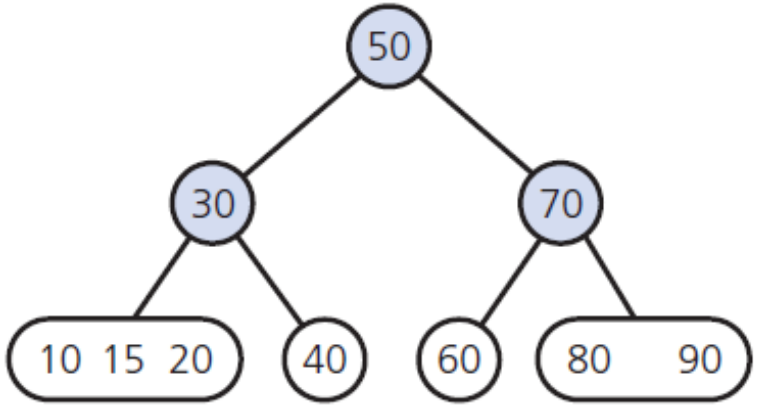
Adding Data to 2-3-4 Trees

The steps for adding 100 to the tree in Figure

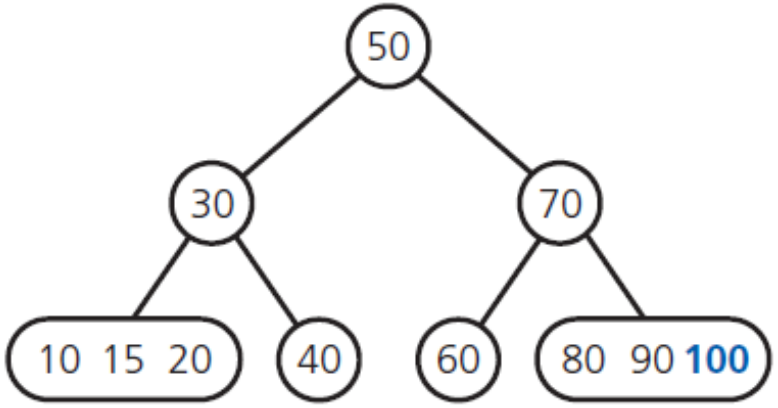
(b) After adding 90 to the root's right child



(a) After splitting the 4-node

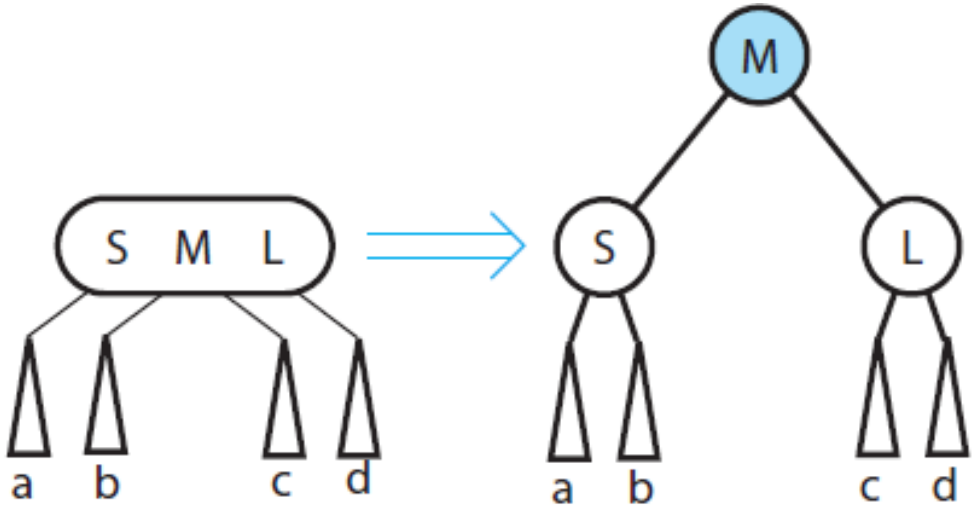


(b) After adding 100 to the rightmost leaf



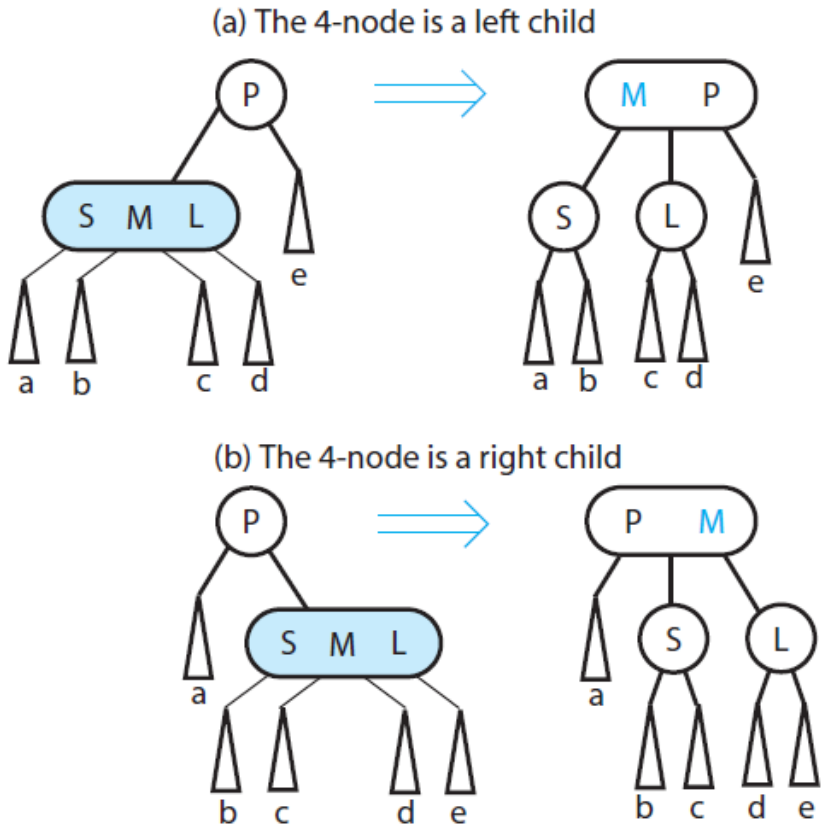
Adding Data to 2-3-4 Trees

Splitting a 4-node root when adding data to a 2-3-4 tree



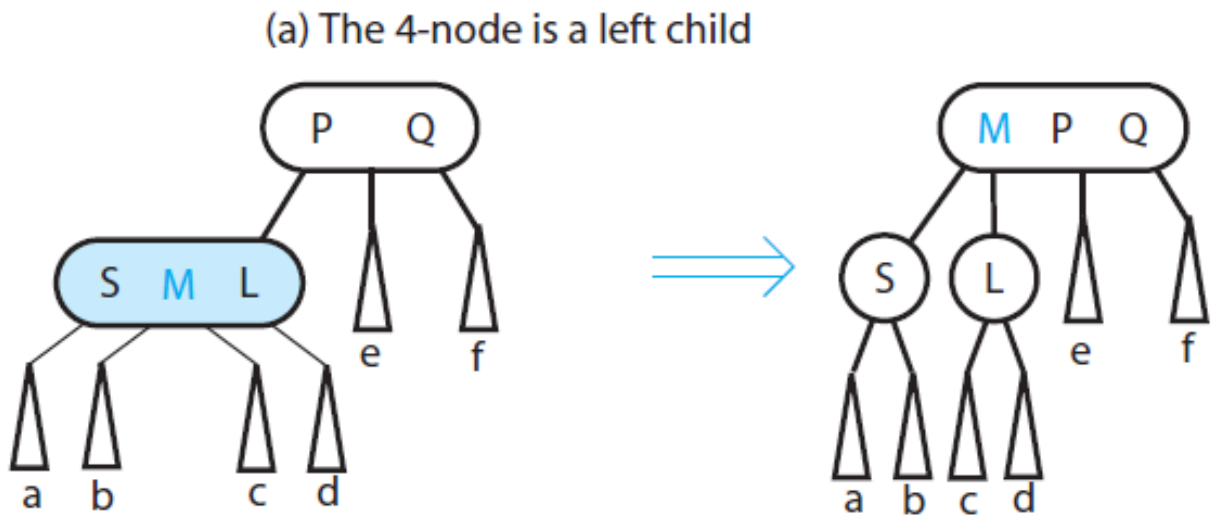
Adding Data to 2-3-4 Trees

Splitting a 4-node whose parent is a 2-node when adding data to a 2-3-4 tree



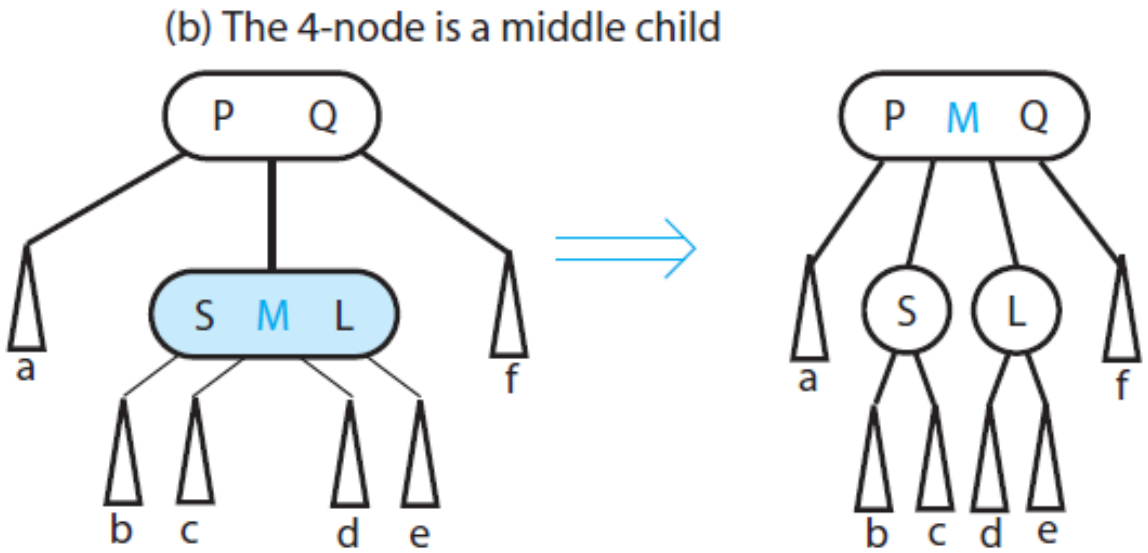
Adding Data to 2-3-4 Trees

Splitting a 4-node whose parent is a 3-node when adding data to a 2-3-4 tree



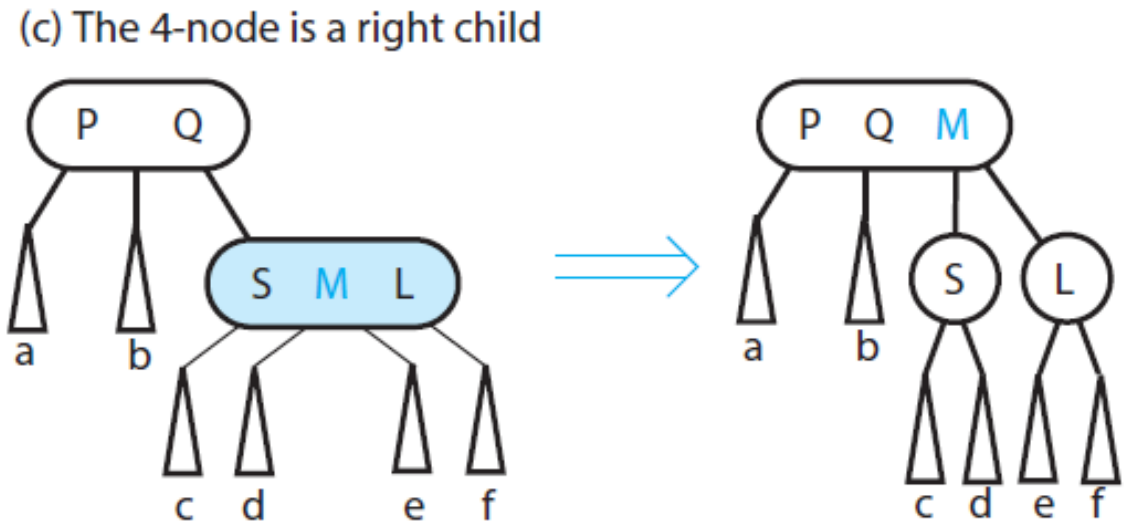
Adding Data to 2-3-4 Trees

[Continued]



Adding Data to 2-3-4 Trees

[Continued]



Removing Data from a 2-3-4 Tree

- Has same beginning as removal algorithm for a 2-3 tree
- Transform each 2-node into a 3-node or a 4-node
- Insertion and removal algorithms for 2-3-4 tree require fewer steps than for 2-3 tree

Thank you