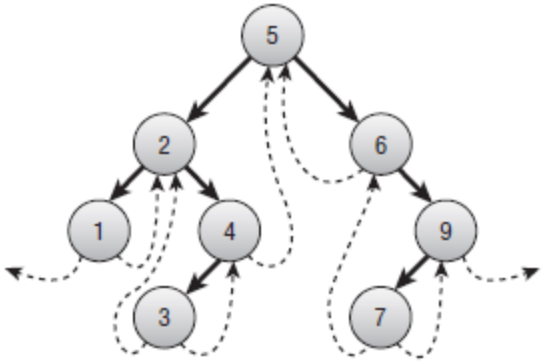
Trees, Part 2

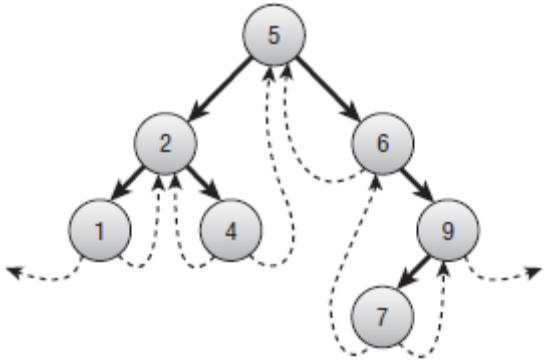


Agenda

- Threaded Trees
- Specialized Tree Algorithms
- Summary
- Exercises

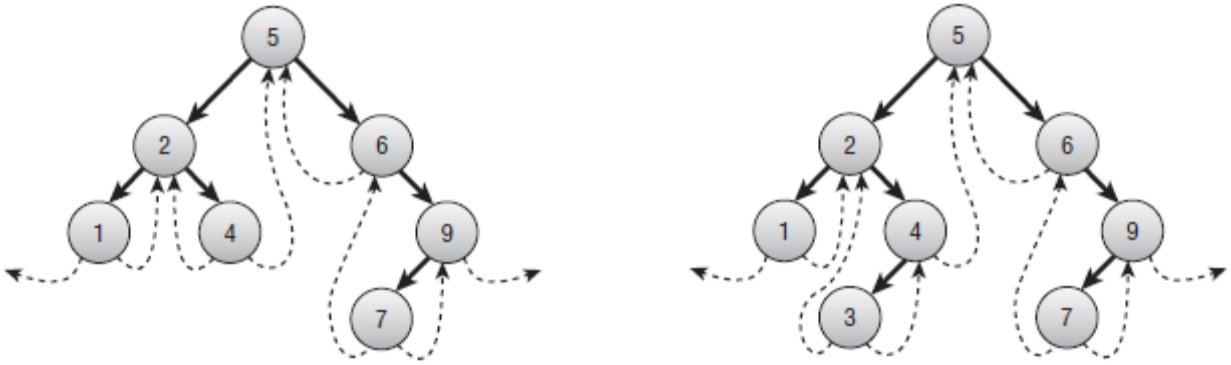
Threaded Trees

- Threads let you move through the nodes in a tree in unusual ways
- The most common kind is a symmetrically threaded tree, which helps you perform inorder and reverse inorder traversals



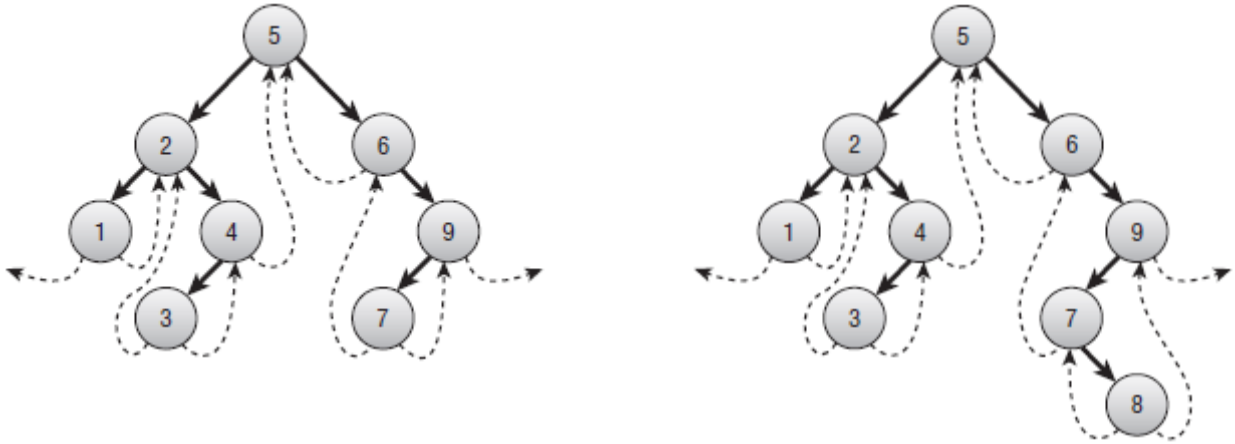
Inserting Left Children

- When you insert a node as a left child, its left thread points where the parent's left thread used to point



Inserting Right Children

- When you insert a node as a right child, its right thread points where the parent's right thread used to point



Using Threads

```
InorderWithThreads(BinaryNode: root)
    // Start at the root.
    BinaryNode: node = root

    // Remember whether we got to a node via a branch or thread.
    // Pretend we go to the root via a branch so we go left next.
    Boolean: via_branch = True

    // Repeat until the traversal is done.
    While (node != null)
        // If we got here via a branch, go
        // down and to the left as far as possible.
        If (via_branch) Then
            While (node.LeftChild != null)
                node = node.LeftChild
            End While
        End If
```

Using Threads

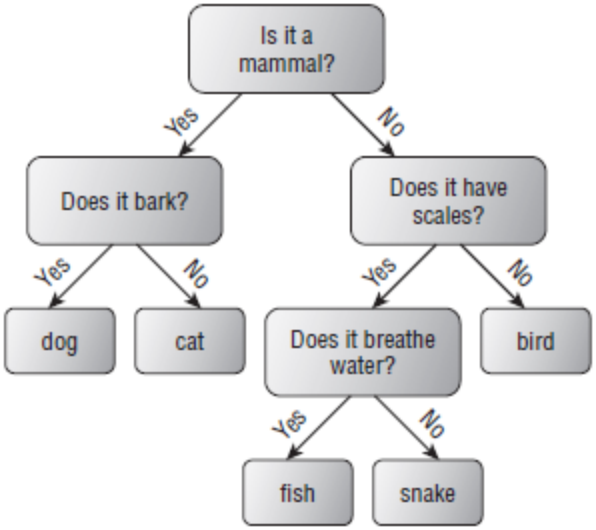
```
        // Process this node.
        <Process node>

        // Find the next node to process.
        If (node.RightChild == null) Then
            // Use the thread.
            node = node.RightThread
            via_branch = False
        Else
            // Use the right branch.
            node = node.RightChild
            via_branch = True
        End If
    End While
End InorderWithThreads
```

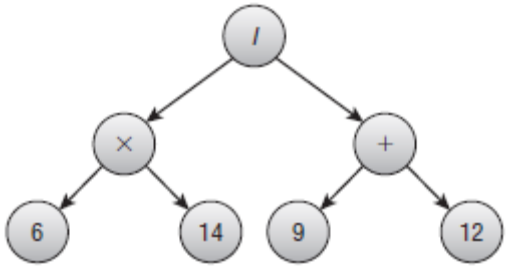
Specialized Tree Algorithms

- The Animal Game
- Expression Evaluation
- Interval Trees
- Quadtrees
- Tries

The Animal Game



Expression Evaluation

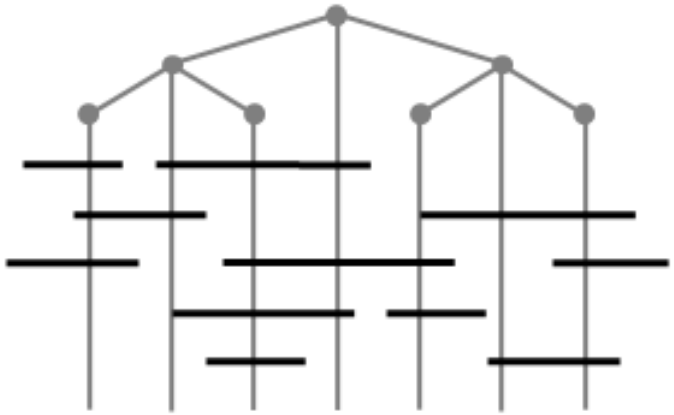


Expression Evaluation

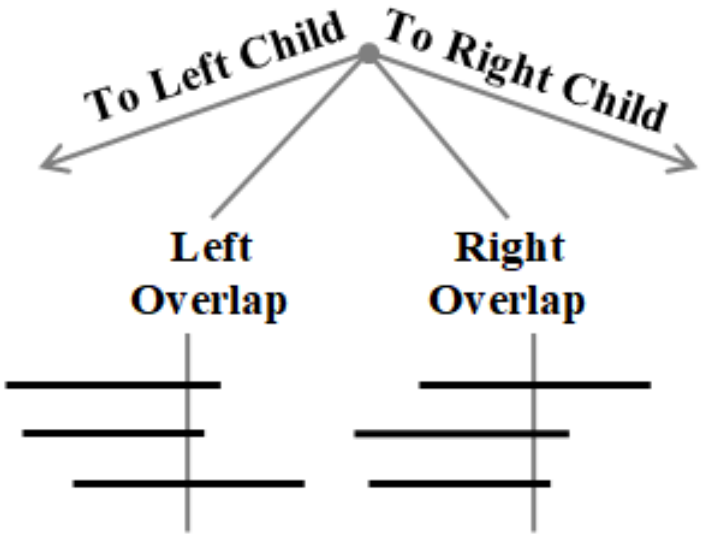
```
Class ExpressionNode
    Operators: Operator
    ExpressionNode: LeftOperand, RightOperand
    String: LiteralText

    // Evaluate the expression.
    Float: Evaluate()
        Case Operator
            Literal:
                Return Float.Parse(LiteralText)
            Plus:
                Return LeftOperand.Evaluate() + RightOperand.Evaluate()
            Minus:
                Return LeftOperand.Evaluate() - RightOperand.Evaluate()
            ...
        End Case
    End Evaluate
End ExpressionNode
```

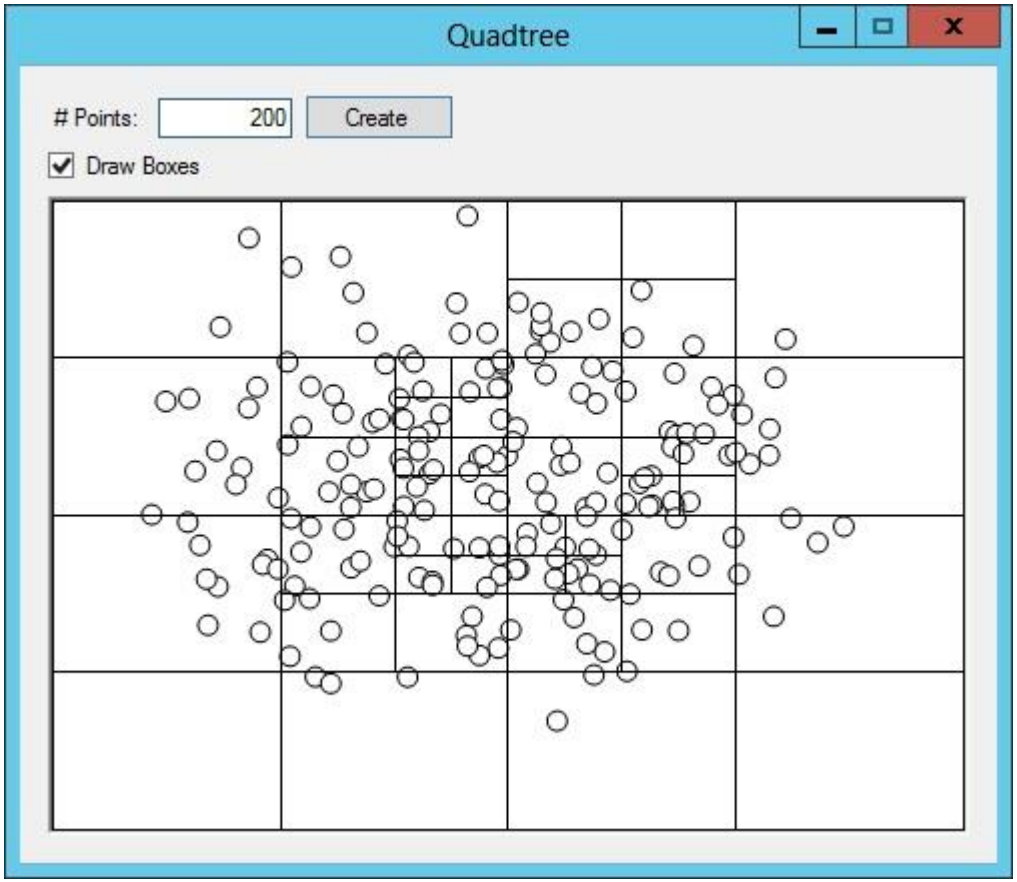
Interval Trees



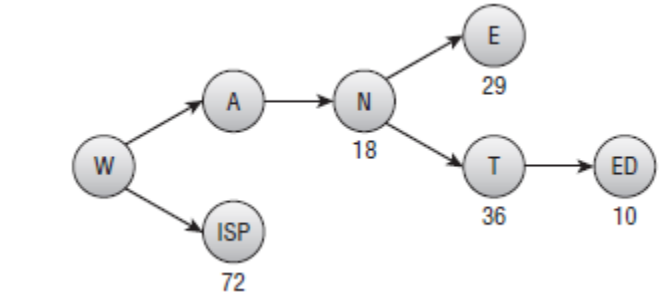
Interval Trees

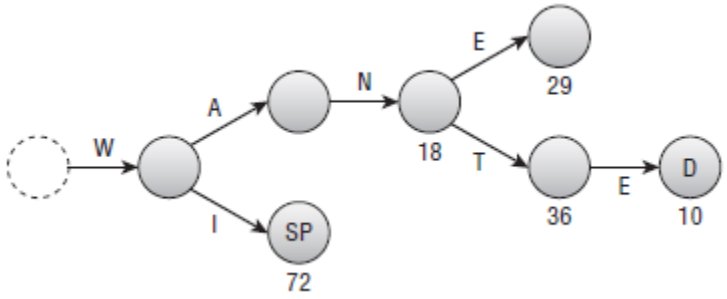


Quadtrees



Tries





Summary

- Threaded Trees
- Specialized Tree Algorithms
  - The Animal Game
  - Expression Evaluation
  - Interval Trees
  - Quadtrees
  - Tries

Exercises

- Chapter 10 Exercises 16, 17, 19 – 44 non-starred problems.
- Bonus: Chapter 10 Exercises 18 – 44 starred problems.
- Read *Essential Algorithms, 2e* Chapter 11 pages 349 – 358. (Stop before the section “B-Trees.”)