CSCI 136
Data Structures &
Advanced Programming

Jeannie Albrecht
Lecture 22
April 14, 2014

Administrative Details

• Lab 7 due today
  • Any questions?
• Handout: Lab 8
  • Ideally you should bring LexiconNode design doc to lab so we can discuss at beginning
  • LexiconNode is the recursive data structure
  • LexiconTrie manipulates LexiconNodes
  • LexiconNode at root! Use ‘ ’ (single blank space) character

Last Time

• Looked at binary expression trees
• Began talking about how to implement binary trees in Java
  • Defined weird “empty” trees
  • Defined three constructors

Today’s Outline

• Continue discussing trees
  • Implement BinaryTree
  • Prove tree properties
  • Take a closer look at decision trees
  • Learn how to traverse trees

BinaryTree Recap

• BinaryTree class
  • Instance variables
    • BT parent, BT left, BT right, Object value

Implementing BinaryTree

• Methods (on board last time – see Ch 12 for more info):
  • (All “left” methods have equivalent “right” methods)
  • public BinaryTree()
    • // generates an empty node (EMPTY)
      • // parent and value are null, left=right=ths
  • public BinaryTree(E value)
    • // generates a tree with a non-null value and two empty (EMPTY) subtrees
  • public BinaryTree(E value, BinaryTree<E> left, BinaryTree<E> right)
    • // returns a tree with a non-null value and two subtrees
  • public void setLeft(BinaryTree<E> newLeft)
    • // sets left subtree to newLeft
  • protected void setParent(BinaryTree<E> newParent)
    • // calls from setLeft and setRight to keep all “links” consistent
Implementing BinaryTree

- Methods:
  - public BinaryTree<E> left(): returns left subtree
  - public BinaryTree<E> parent(): returns reference to parent node, or null
  - public boolean isLeftChild(): returns true if this is a left child of parent
  - public E value(): returns value associated with this node
  - public void setValue(E value): sets the value associated with this node
  - public Iterator<E> iterator(): returns an in-order iterator of the elements

BT Methods

- Other useful methods to consider
  - size(): number of descendants
  - height(): height of node in tree

Left as an exercise…think about these. How would they be defined?

An aside: visualizing binary trees

BT Questions/Proofs

- Prove that number of nodes at level n <= 2^n.

- Prove that number of nodes in tree of height n is <= 2^(n+1) - 1.

Representing Knowledge

- Trees can be used to represent knowledge
  - Example: InfiniteQuestions game

- We often call these trees decision trees
  - Leaf: object
  - Internal node: question to distinguish objects

- Move down decision tree until we reach a leaf node

- Check to see if the leaf is correct
  - If not, add another question, make new and old objects children

Decision Trees

- Applications
  - InfiniteQuestions game
  - Medical diagnosis

- Issues with decision trees
  - How do we pick the right questions?
    - We want fewest number of questions on average path through tree, which highest confidence of obtaining correct answer
  - What problems occur when we pick the wrong questions?

Building Decision Trees

- Gather/obtain data

- Run correlation analysis
  - Make greedy choices: Find good questions that divide data into halves (or as close as possible)

- Construct tree with shortest height

- Example