Lecture 6.45

Applications of Binary Trees (Continued)

Agenda for Today's Lecture

- Recap: Hierarchical Data Representation
- Expression Trees: Structure and Evaluation
- Huffman Coding Trees: Data Compression
- Binary Tries in Network Routing
- Decision Trees in Machine Learning
- Syntax Trees in Compilers

Recap: The Power of Binary Trees

- Binary Trees restrict nodes to a maximum of two children (Left and Right).
- Provides an ideal framework for Divide and Conquer algorithms.
- Previously covered:
- - Hierarchical file systems.
- - Binary Search Trees (BST) for fast lookup operations (O(log n)).

Expression Trees: Representing Math

- An Expression Tree is a specific kind of binary tree used to represent expressions.
- Leaves are operands (constants or variables like 3, 5, x, y).
- Internal nodes are operators (+, -, *, /).
- The structure naturally enforces operator precedence and associativity.
- Does not require parentheses to evaluate correctly.

Evaluating an Expression Tree

- Evaluation relies on a Post-order Traversal (Left-Right-Root).
- Algorithm Step-by-Step:
- 1. Recursively evaluate the left subtree.
- 2. Recursively evaluate the right subtree.
- 3. Apply the operator at the root to the results of step 1 and 2.
- Example output generates Postfix notation inherently.

Huffman Coding: Optimal Data Compression

- Huffman Coding is a lossless data compression algorithm.
- It assigns variable-length binary codes to input characters based on their frequencies.
- Most frequent characters get the shortest codes.
- Uses a Full Binary Tree (Huffman Tree) to generate these prefix codes.
- Ensures no code is a prefix of another, allowing unambiguous decoding.

Constructing a Huffman Tree

- 1. Create a leaf node for each character and build a min-heap of all leaves.
- 2. Extract the two nodes with the lowest frequencies.
- 3. Create a new internal node with a frequency equal to the sum of the two nodes.
- 4. Make the first extracted node its left child (edge $= 0$) and the second its right child (edge $= 1$).
- 5. Insert the new node back into the heap.
- 6. Repeat until only one node remains (the root).

Binary Tries for Network Routing

- A Trie (Prefix Tree) can be binary when the alphabet is just $\{0, 1\}$.
- Widely used in Internet Routers to store routing tables.
- Enables Longest Prefix Matching (LPM) for IP addresses.
- Left branch represents bit '0', Right branch represents bit '1'.
- Allows O(W) lookup time, where W is the length of the IP address (e.g., 32 bits for IPv4).

Decision Trees in Machine Learning

- A predictive modeling approach used in classification and regression.
- Internal nodes represent a 'test' on an attribute (e.g., 'Is age ¿ 30?').
- Branches represent the outcome of the test (True/False or Yes/No).
- Leaf nodes represent the final class label or decision.
- Often strictly binary in implementations like CART (Classification and Regression Trees).

Abstract Syntax Trees (AST) in Compilers

- Used by compilers (including Python's interpreter) to understand source code.
- ASTs represent the syntactic structure of the code.
- Strips away formatting details (like brackets, semicolons) keeping only the logic.
- Many sub-components of an AST are binary (e.g., BinaryOperations like assignments or comparisons).
- Crucial for code optimization and code generation.

Summary of Advanced Applications

- Binary trees are far more than just BSTs for sorting.
- Expression Trees evaluate mathematics natively.
- Huffman Trees enable optimal lossless data compression.
- Binary Tries are the backbone of IP packet routing.
- Decision Trees provide readable machine learning models.
- Syntax Trees are how computers understand programming languages.

Looking Ahead & Next Steps

- This concludes Unit 6: Binary Trees.
- Ensure you complete the practical assignments on BST operations.
- Review all tree traversal methods (Inorder, Preorder, Postorder) for the upcoming exam.
- Next time: We will begin reviewing comprehensive course topics and practical interview questions on Data Structures.