CHAPTER 6 TREES
PART 2

Huffman Tree, Binary Search Tree (Basics)

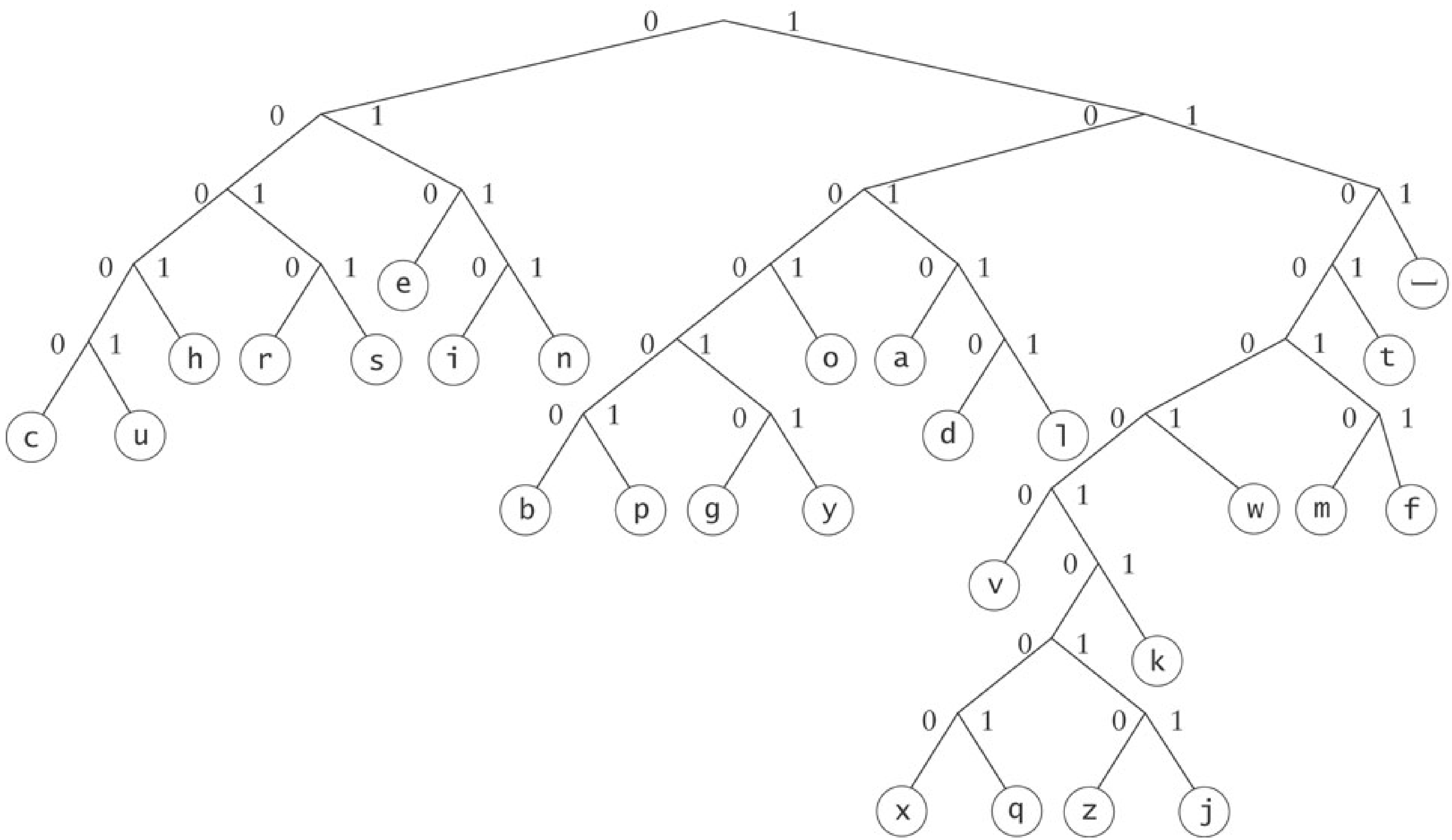
Key Topics

- Huffman Tree
  - Definition & Usage
  - Encode
  - Decode
- Binary Search Tree
  - Definition
  - Search
  - Add

Huffman Tree

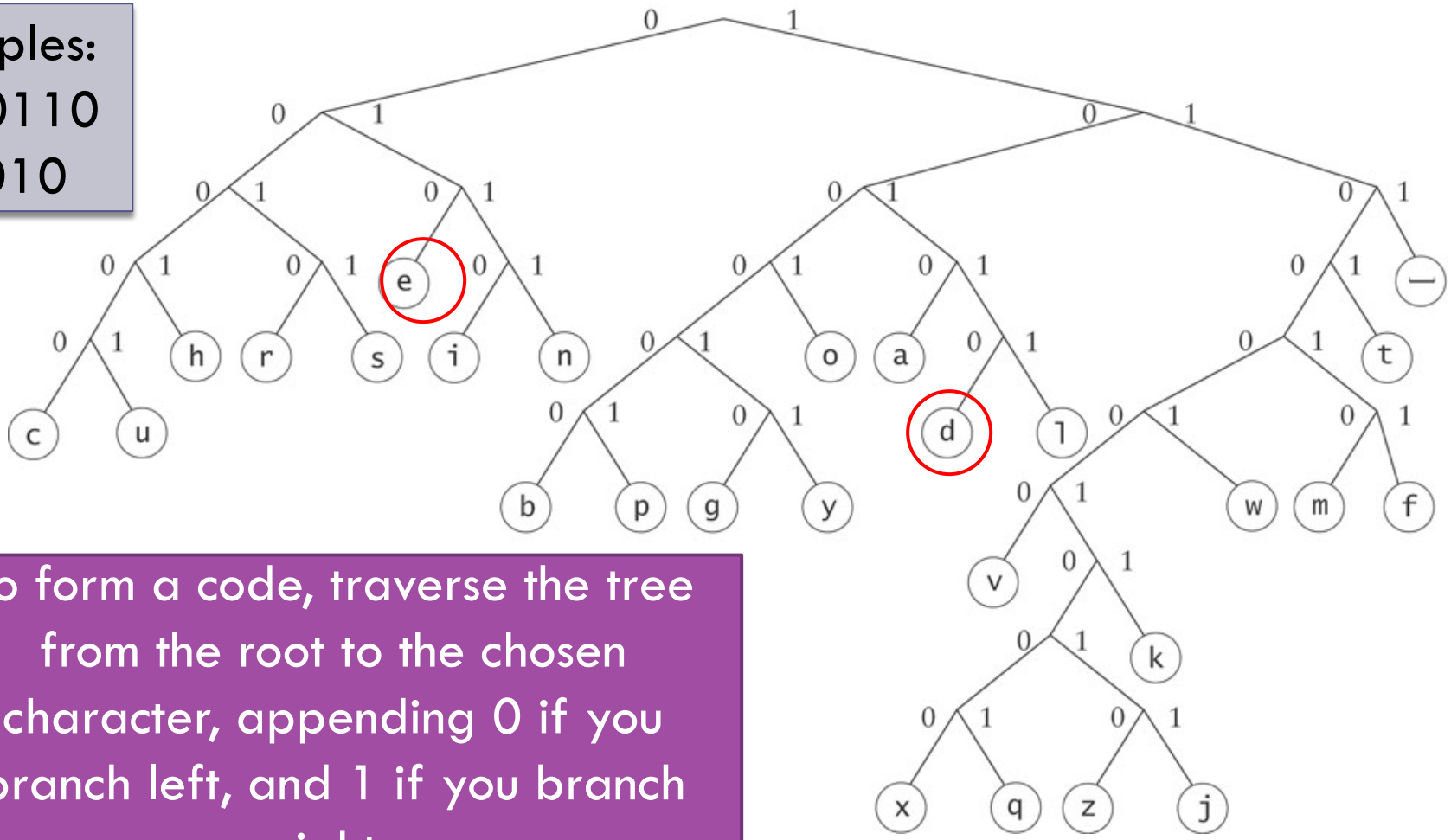
- Represents ***Huffman codes*** for characters in a text file
- Huffman code (Unlike ASCII or Unicode)
  - Use **different number of bits to encode letters**
  - Use **fewer** bits for **more common** characters
    - Fewer total number of bytes than ASCII or Unicode
    - Smaller files and less storage requirements
  - Used in many programs that **compress files**

Huffman Tree - Example



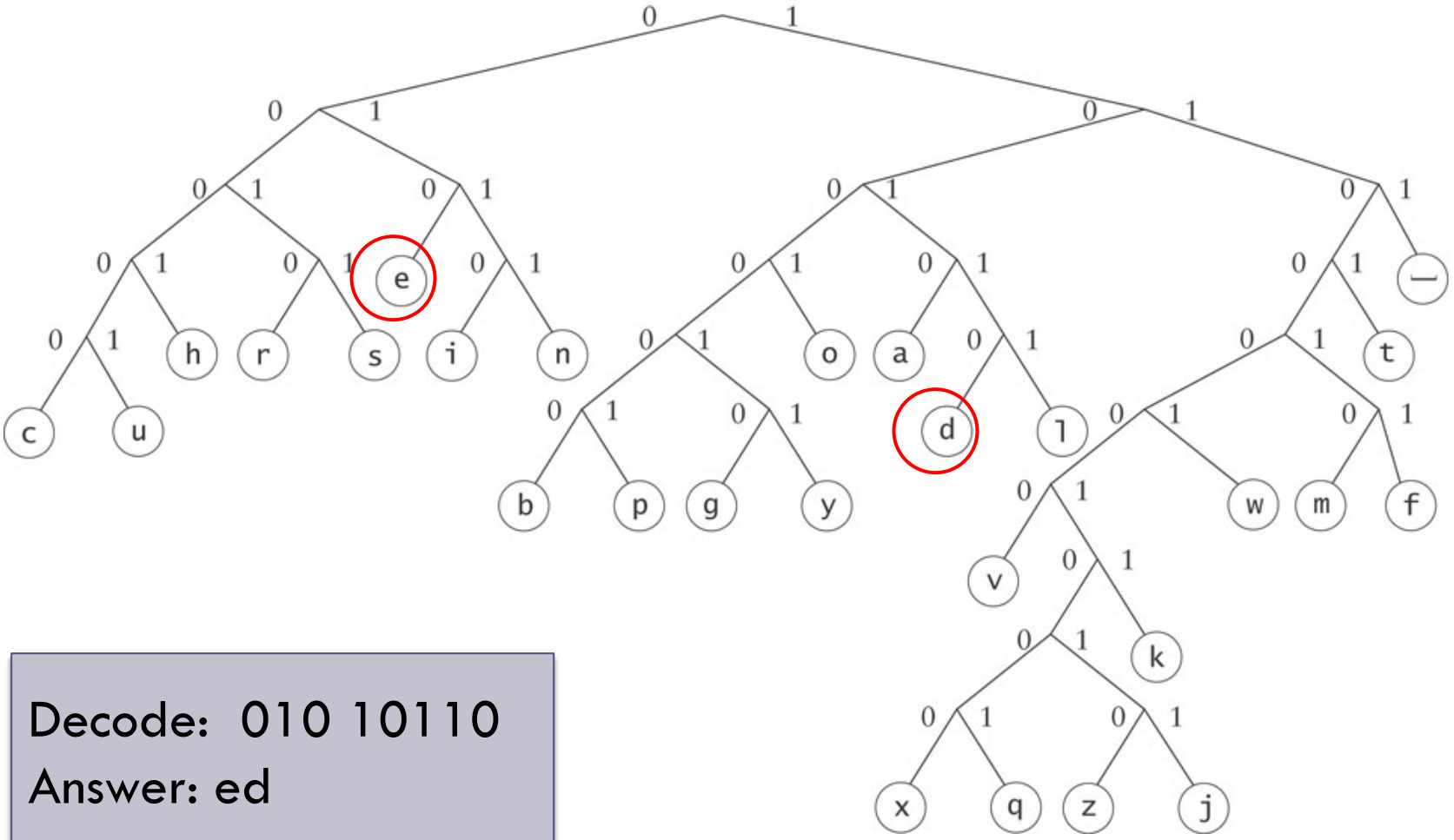
Huffman Tree – Example & Encode

Examples:
d : 10110
e : 010



To form a code, traverse the tree from the root to the chosen character, appending 0 if you branch left, and 1 if you branch right.

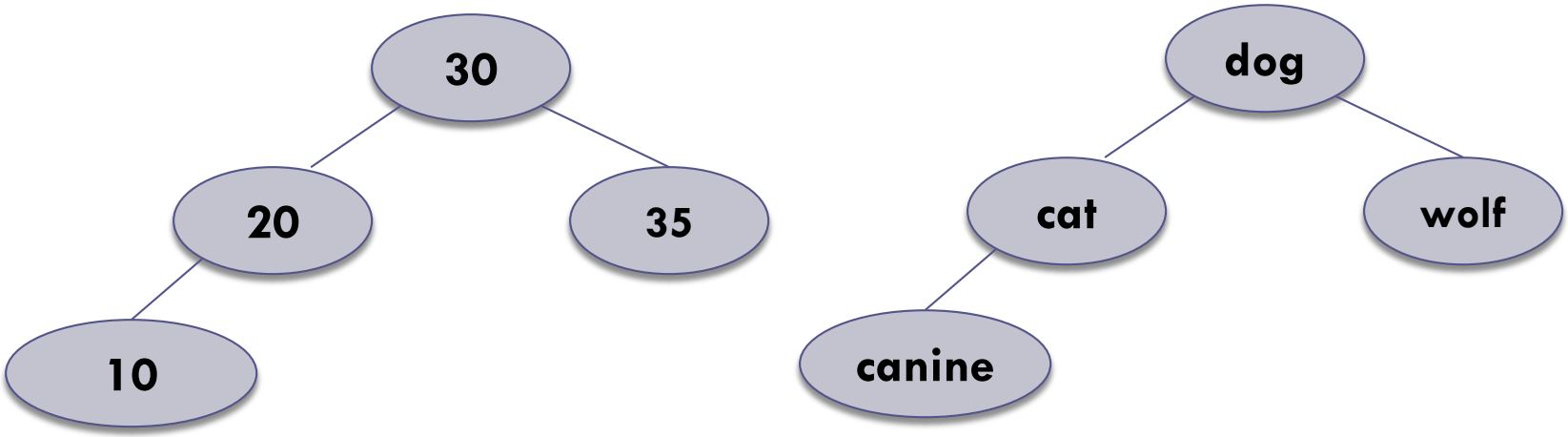
Huffman Tree – Example & Decode



Decode: 010 10110

Answer: ed

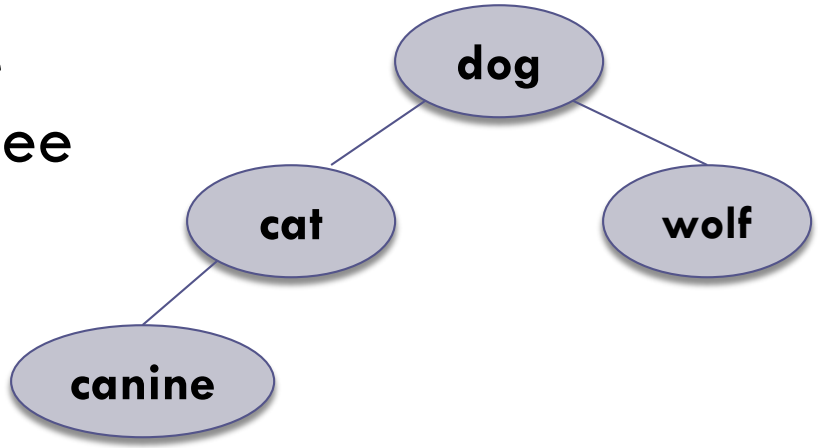
Binary Search Tree - Definition



- **Binary Search Tree = Binary Tree + data restriction:**
  - Each node's left subtree TL has values < node's value
  - Each node's right subtree TR has values > node's value

Binary Search Tree - Definition

- **All elements** in the **left** subtree **precede** those in the **right** subtree



- A formal definition:

A set of nodes T is a binary search tree if either of the following is true:

- T is empty
- If T is not empty, its root node has two subtrees, T_L and T_R, such that T_L and T_R are binary search trees and the value in the root node of T is greater than ALL values in T_L and is less than ALL values in T_R

Binary Search Tree – Search (Recursive)

if the tree is empty

    return null *(target is not found)*

else if the target matches the root node's data

    return the data stored at the root node

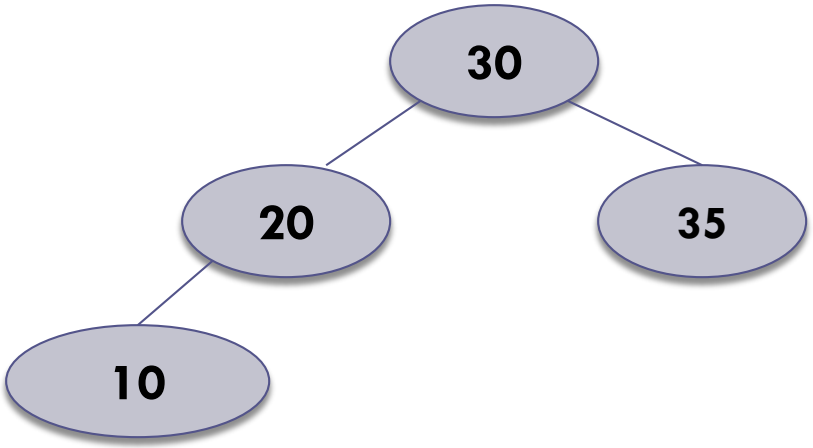
else if the target is less than the root node's data

    return the result of searching the left subtree of the root

else

    return the result of searching the right subtree of the root

Binary Search Tree – Search Example



- Targets: 20, 50

Binary Search Tree – Add (Recursive)

if the tree is empty

    empty tree is insertion point, replace empty tree by newItem

else if the newItem matches the root node's data

    return false (duplicate)

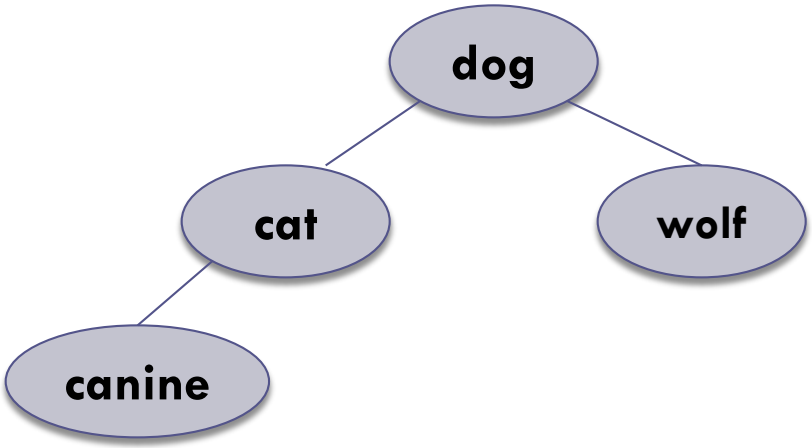
else if the newItem is less than the root node's data

    return the result of addng the left subtree of the root

else

    return the result of adding the right subtree of the root

Binary Search Tree – Add Example



After adding panda: